Inter-Office Memorandum

To	Communication Protocols	Date	February 8, 1980
From	Ed Taft	Location	Palo Alto
Subject	Miscellaneous Services (Edition 4)	Organization	PARC/CSL

XEROX

Filed on: <Pup>MiscServices.bravo

A number of hosts, including Maxcs, Gateways, and IFSs, support a collection of simple, miscellaneous services. These services include only those functions that can be completed in a single exchange of Pups.

It is conceded that the organization of these services and the syntax of requests and responses is somewhat haphazard. This arrangement is due for a thorough overhaul. Most likely these services will be re-implemented on top of the Remote Procedure Call mechanism now under development.

This memo supercedes most of Pup Servers on Maxc'' of October 16, 1975. The other protocols mentioned therein (Telnet, Gateway Information, FTP, and Echo) are now described in separate documents.

Edition 2: Laurel-style Mail Check Request added.

Edition 3: Address lookup request added.

Edition 4: Validate Recipient request added, and Mail Check and Authenticate extended to include optional registry qualification.

General

A user requests one of these services by sending a Pup to socket 4 on a server system. The server performs the requested operation and generates a reply Pup. Since either the request or the reply could be lost, the user process must be prepared to retransmit the request if necessary, and the services provided must be of the sort that do not have incremental effects if requested repeatedly. Generally, these functions are simply requests for information.

The operation to be performed is determined by the Type field of the incoming Pup. In some cases, the reply is similarly distinguished by the Pup Type. In the following descriptions, the Pup Type assignments are given in octal. The Pup ID of the request may be selected arbitrarily by the user, and the Pup ID of the reply will match it. Users are advised to select Pup IDs in a manner that will enable them to distinguish between the replies to successive requests, if necessary.

Except where otherwise noted, a *string* is simply a packed array of bytes carried in the data portion of a Pup. A Pup carrying a string contains no other data.

Individual Services

Date and Time

The present date and time may be requested in one of four different formats. Not all servers necessarily support all four formats. The one that is universally supported by existing servers is the new GMT-based Alto time standard.

> *Pup Type:* String time request = 200B
> *Pup Contents:* None.
>
> *Pup Type:* String time reply = 201B
> *Pup Contents:* A string consisting of the current date and time in the form   11-SEP-75 15:44:25''.
>
> *Pup Type:* Tenex time request = 202B
> *Pup Contents:* None.
>
> *Pup Type:* Tenex time reply = 203B
> *Pup Contents:* 6 bytes containing the date and time in Tenex internal format (see Tenex Jsys manual). The leftmost 18 bits (the date) are right-justified in the first 3 bytes (treating it as a 24-bit integer), and the rightmist 18 bits (the time) in the second 3 bytes. *This standard is obsolete and should no longer be used.*
>
> *Pup Type:* Alto time request (old standard) = 204B
> *Pup Contents:* None.
>
> *Pup Type:* Alto time response (old standard) = 205B
> *Pup Contents:* 4 bytes containing the date and time as a 32-bit number in Alto internal format, which is the number of seconds since midnight, January 1, 1901, local time. *This standard is obsolete and is no longer supported.*
>
> *Pup Type:* Alto time request (new standard) = 206B
> *Pup Contents:* None.
>
> *Pup Type:* Alto time response (new standard) = 207B
> *Pup Contents:* 10 bytes in all, organized as 5 16-bit words:

words 0, 1	Present date and time: a 32-bit integer representing number of seconds since midnight, January 1, 1901, Greenwich Mean Time (GMT).
word 2	Local time zone information. Bit 0 is zero if west of Greenwich and one if east. Bits 1-7 are the number of hours east or west of Greenwich. Bits 8-15 are an additional number of minutes. Most time zones are an integer number of hours from Greenwich, so the minutes field will ordinarily be zero.
word 3	Day of the year on or before which Daylight Savings Time takes effect locally, where 1 = January 1 and 366 = December 31. (The actual day is the next preceding Sunday.) The correspondence between numbers and days is based on a leap-year.
word 4	Day of the year on or before which Daylight Savings Time ends. If Daylight Savings Time is not observed locally, both the start and end dates should be 366.

The local time parameters in words 2 through 4 are those in effect at the server's location. It is the user's responsibility to determine whether or not it is appropriate to apply them

locally.

Mail Check

This service permits a user to check for the arrival of new mail relatively cheaply. For example, on Maxc the cost is at least an order of magnitude less than that for invoking the Mail Transfer Protocol. However, since the cost is not zero and there are potentially many simultaneous users of this service, programs that generate periodic mail check requests automatically should do so no more frequently than once every 5 minutes.

The Mail Check protocol has two variants, *Msg-style* and *Laurel-style*. The former returns the   New mail exists'' reply only if the mailbox file has been written more recently than it was last read, whereas the latter requires only that the mailbox be non-empty.

> *Pup Type:* Mail check request (Msg-style) = 210B
> *Pup Contents:* A string specifying the mailbox name.
>
> *Pup Type:* Mail check request (Laurel-style) = 214B
> *Pup Contents:* A string specifying the mailbox name.
>
> *Pup Type:* New mail exists reply = 211B
> *Pup Contents:* Optional human-readable text.
>
> *Pup Type:* No new mail exists reply = 212B
> *Pup Contents:* None.
>
> *Pup Type:* No such mailbox reply = 213B
> *Pup Contents:* A human-readable explanation.

The mailbox name in the Mail Check request may optionally be qualified by a registry name, i.e., in the form *simpleName.registry*. If a registry name is present, the server should verify that it corresponds to a true mail registry (socket 7) on the same machine, and reject the request with a No Such Mailbox reply if the registry is improper.

Network Directory Lookup

This service translates between inter-network name expressions and address values. See the memo   Naming and Addressing Conventions for Pup'' for further information.

The data base for this service presently contains only host names (Altos and other machines) and is centrally maintained. This facility will eventually be replaced by a distributed Clearinghouse service that permits storing other types of information.

> *Pup Type:* Name lookup request = 220B
> *Pup Contents:* A string consisting of an inter-network name expression.
>
> *Pup Type:* Name lookup response = 221B
> *Pup Contents:* One or more 6-byte blocks containing the address(es) corresponding to the name expression. Each block is a Pup *Port* structure, with the network and host numbers in the first two bytes and the socket number in the last four bytes.
>
> *Pup Type:* Address lookup request = 223B
> *Pup Contents:* A Port (6 bytes).
>
> *Pup Type:* Address lookup response = 224B
> *Pup Contents:* A string consisting of an inter-network name expression that matches the request Port. *This service is not provided by all directory lookup servers at the present time.*

Pup Type: Directory lookup error reply = 222B
Pup Contents: A human-readable explanation.

Where is User

This is a Maxc-specific service that gives out information about jobs belonging to a particular user. This permits programs such as Chat to determine whether it is appropriate to attach to an existing job or log in a new one. IFS servers also implement a degenerate form of this protocol.

Pup Type: Where is user request = 230B
Pup Contents: A string specifying the user name.

Pup Type: Where is user response = 231B
Pup Contents: Zero or more pairs of bytes designating jobs in the server machine belonging to the specified user. In each pair, the first byte is a Tenex job number and the second is a Tenex terminal number (377B indicates that the job is detached). By convention, job numbers are represented externally in decimal but terminal numbers in octal.

Pup Type: Where is user error reply = 232B
Pup Contents: A human-readable explanation.

Network Directory Update

This service implements part of the Network Directory Update protocol, which is the means by which the centrally-maintained Network Directory data base is distributed to servers. The protocol is described fully in a separate document, Pup Network Directory Update Protocol''. Here we simply list the Miscellaneous Services Pup Types that are used:

Pup Type: Network directory version information = 240B

Pup Type: Network directory update request = 241B

Alto Boot and Boot Update

The protocols by means of which Altos are boot-loaded over the Ethernet and boot files are kept up-to-date are described in a separate document, Alto Boot Protocol''. Here we simply list the Miscellaneous Services Pup Types that are used:

Pup Type: Send boot file request = 244B

Pup Type: Boot directory request = 245B

Pup Type: Boot directory reply = 246B

User Authentication

This is a rudimentary authentication service that simply checks the validity of user name and password combinations on the server machine. It is somewhat specialized to the short-term needs of the Laurel system. It will be replaced eventually by a more secure authentication mechanism based on encryption.

Pup Type: Authenticate request = 250B
Pup Contents: Two Mesa strings (more precisely, *StringBody*s), packed in such a way that the *maxLength* of the first string may be used to locate the second string. The first string is a user name and the second a password.

Pup Type: Authenticate positive response = 251B
Pup Contents: None. This response indicates that the name/password combination is valid.

Pup Type: Authenticate negative response = 252B
Pup Contents: A human-readable explanation.

Note that only the *Authenticate request* contains Mesa strings. This is done because the normal convention for strings does not provide for packing multiple strings in one packet.

The user name in the Authenticate request may optionally be qualified by a registry name, i.e., in the form *simpleName.registry*. If a registry name is present, the server should verify that it corresponds to a true mail registry (socket 7) on the same machine, and reject the request with an Authenticate negative response if the registry is improper.

Validate Recipient

This protocol is used to validate a recipient in a manner similar to what a mail server does with a Mailbox property, but using a single packet exchange. This is a temporary protocol, intended to facilitate coexistence of new and old Mail protocols.

Pup Type: Validate recipient request: 266B
Pup Contents: a Mesa string containing a recipient name optionally qualified by   .registry''.
The registry, if present, is expected to be one actually located at the server.

Pup Type: Validate recipient Yes reply: 267B
Pup Contents: none

Pup Type: Validate recipient No reply: 270B
Pup Contents: none

A Yes response means that the server would be willing to accept mail for that recipient, and a No response means that the server would reject mail for that recipient, or that the specified registry is not located at the server. If the server cannot conveniently determine whether or not it would accept mail for a recipient, it should answer Yes.

It is permissible for the server occasionally to reply Yes erroneously, but never to reply No erroneously. An erroneous Yes might result in the delivery of messages that would later be rejected and returned to the sender, but an erroneous No would altogether prevent delivery to a valid recipient.

Note: the principal intended client of this protocol is the new DMS mail server, which maintains a large local cache of validated recipient names. Hence Validate Recipient requests are expected to be dynamically infrequent except when a DMS mail server is restarting after a crash. This is important, since each such request incurs a full per-packet processing cost in addition to the cost of the validation operation itself. This is in contrast to the recipient validation portion of the Mail Transfer Protocol, where an entire recipient list is presented at once (in a single FTP command).

Implementation Status

The following table summarizes the services routinely available from server hosts at the present time.

Service	Maxc	Nova Gateway	Alto Gateway	IFS
Date and Time (String)	X			
Date and Time (Tenex)	X	X		
Date and Time (old Alto)		X		

Date and Time (new Alto)	X	X	X	X
Mail Check	X			X
Directory Name Lookup	X	X	X	X
Directory Address Lookup			X	X
Where is User	X			X
Network Directory Update	X	X	X	X
Alto Boot		X	X	X
Authenticate	X			X
Validate Recipient	X			X