Palo Alto Research Center

Transport of Electronic Messages Through a Network

By Roy Levin and Michael D. Schroeder

XEROX

Transport of Electronic Messages Through a Network

by Roy Levin and Michael D. Schroeder

CSL-79-4 APRIL 1979

Abstract: We list design objectives for a distributed mechanism to transport digital memoranda in a network, and discuss the associated administrative functions. We examine registering, authenticating, locating, and grouping users; define name mappings associated with message delivery; and consider the distribution of services among the computing elements in a network. Based on these analyses, we outline the structure for a distributed transport mechanism.

CR Categories: 3.57, 3.81, 4.9

Key words and phrases: electronic mail, computer networks, protocols, message systems

XEROX
PALO ALTO RESEARCH CENTER
3333 Coyote Hill Road / Palo Alto / California 94304

1. Introduction

Message systems that communicate memoranda among members of a community are an important application of computers and networks. Such message systems generally rely on a central facility used by all members of the community, such as a time sharing system, to transport messages from one user to another. While a centralized approach simplifies the design of the message transport mechanism, it complicates use, growth, and administration when the community is large, dispersed, and includes people from multiple organizations. A distributed transport mechanism is more complex to design, but permits more natural use, simple growth, and flexible administration.

One approach to distribution is exemplified by the Arpanet message system protocols [ArpaNet] that allow multiple local message systems to be interconnected and messages to be forwarded from one site to another. Unfortunately, recipient names contain explicit routing and resource information that may change when additional computers are added within a local community. Also, the potential reliability of a distributed system is not realized, for when a user's local message system computer fails that user is cut off from message service. In this paper we sketch the design for a distributed message transport mechanism that decouples naming from routing and resource management, and that continues to provide all users with message service when individual computers fail.

1.1 Environment

Figure 1 illustrates the network environment in which we consider transport of messages. This environment exists within the Xerox research and development community in the United States. Most computing is done in personal computers, labeled PC in the figure, that each have a small removable disk for local storage of files. These computers may be used by various persons at different times for different tasks. The personal computers are interconnected by a network that consists of Ethernet packet broadcast local networks [Metcalfe and Boggs], gateways, and telephone lines. Also connected to the network are shared computers, called *servers*, that provide various services to the community, including file storage, printing, and name lookup. The network is arranged so that high bandwidth local networks interconnect computers in each local area, while lower bandwidth telephone lines link these areas.

Reading, filing, and indexing of received messages and preparation of messages to be sent are performed by a user at a personal computer, perhaps with the assistance of filing and printing servers. Messages are transported among users using the network and various message servers. In the discussion that follows we assume the existence of low-level communication protocols for establishing connections between computers attached to the network and for transferring bits reliably over these connections.

1.2 Role of the transport mechanism

The transport mechanism accepts two basic requests: deliver an uninterpreted bit string as a message to a specified list of recipients, and retrieve all newly arrived messages for a particular user. Given a message to deliver, the transport mechanism must compute where to deliver the message for each named recipient, move the message to that location, buffer it, and finally present the buffered messages to a recipient when asked.

2. Design Goals

This section presents a set of design goals for a distributed transport mechanism. We consider the transport mechanism separately from the design of an interactive, personal message manager employed by users at personal computers. In the discussion that follows the term *client* refers to such a message manager program.

2.1 Reliability

A user wants the assurance that the messages he submits, once accepted by the transport mechanism, will either be delivered to the specified recipients or returned to him with an appropriate explanation. Delivery to a "dead letter" office should be used only in exceptional conditions. Thus, the transport mechanism must be robust with respect to both user/client errors (invalid recipient names, protocol violations) and internal difficulties (communication failures, server unavailability, file storage limitations). The transport mechanism must detect these exceptional conditions and respond to them gracefully, with appropriate notification of problems from which it cannot recover completely. As with other transport mechanisms (postal service, telephone system), certain disruptions may make the transport facility temporarily unavailable to some of its users or degrade the service it provides. Except in catastrophic circumstances, however, the transport mechanism should respond to such disruptions in predictable and reasonable ways, and should not lose messages.

2.2 Performance

The performance of the transport mechanism has two separable components: interactive delay and delivery delay. Interactive delay is the time required for the mechanism to respond to the request of a single user, while delivery delay is the the time required for the mechanism to deliver a message from one user to another. We expect the former to be more important to a user than the latter. Reasoning by analogy with the U.S. postal system, we believe that a user is more likely to be annoyed by long lines and slow clerks at a post office window than by the speed with which his letter is delivered. Thus, a design goal for the distributed transport mechanism is rapid

interaction with clients wishing to send and receive messages, possibly at the expense of rapid movement of messages from sender to recipients. (In the network environment described in section 1.1, the delivery time of messages is measured in seconds, not days. An increase in delivery time from 3 to 30 seconds would not significantly affect the users' perception of message delivery time.)

2.3 Distributed Administration

The users of a distributed network are likely to be managed by different administrative authorities. A distributed transport mechanism should support this decentralized administrative structure, thereby minimizing the need for separate groups to operate under mutually agreeable conventions. Individual authorities should be able to control admission of users to the message communication network and the names by which they are known. In doing so, the representative of the administration should be able to interact directly with the transport facility without having to communicate with a central authority.

2.4 Scaling and Reconfiguration

Networks often change in size and interconnection structure. To design a transport mechanism that cannot gracefully accommodate such changes would be disastrous. In particular, the mechanism should comfortably handle the message traffic of a small network with an appropriately small amount of storage and computing resources. Larger networks naturally require additional resources. The transport mechanism should be organized to allow convenient, dynamic alteration of the network configuration, in both the arrangement of clients and the services required by the mechanism. It should be designed to expand flexibly and naturally as additional communities of users are included or as additional supporting services, e.g., new file storage, become available. This design goal implies that the transport mechanism cannot rely on a single, fixed capacity, central service to implement its function. Such a service, if sufficient for a small network, would be inadequate for a large one and, if adequate for a large network, inappropriate for a small one. Finally, it should be easy to reconfigure the transport mechanism to respond to changes in the distribution of message traffic.

2.5 Security

Secure message transmission is an important requirement within certain user communities. The transport mechanism should support the use of conventional encryption techniques even though it may not provide encryption services directly. The implications of this goal are beyond the scope of this paper and will not be discussed. (See [Needham and Schroeder] for a discussion of the kinds of message-passing protocols that the transport mechanism must support.)

3. Naming

Facilities for registering, authenticating, locating, and grouping recipients are an integral part of a message transport mechanism. The goal of distributed administration suggests that a partitioned naming scheme be used to allow autonomous administration of partitions of the user community. We structure the name space for recipients as a two level hierarchy. A recipient name, or *R-name*, has the form NA.SN, where NA is the unique name of a *naming authority*, and SN is a unique *simple name* within that naming authority. Naming authorities are separately administered partitions of the R-name space, and may correspond to organizational, geographic, or other arbitrary partitions that exist in the user community. Naming authorities do not correspond to message server computers in the transport mechanism. The form of an SN is determined by the naming authority and may vary among the user communities.

All names of message system users presented to the transport mechanism in connection with delivery or receipt of messages are R-names. A user may use arbitrary nicknames or abbreviations when interacting with his message manager. This message manager converts such nicknames to R-names using data external to the transport mechanism before presenting the message for delivery. Facilities for mapping persons' names, addresses, and organizations to R-names (as a telephone book maps these personal identifiers to telephone numbers) also may exist, but also are independent of the transport mechanism. Thus, R-names need not be human sensible identifiers; unique numeric codes could be used if suitable "telephone book" facilities exist.

3.1 Registration servers

The message transport mechanism will include one or more *registration servers* for each naming authority. (Multiple servers are intended to make the registration service for a particular naming authority rapidly accessible from several local areas of the network. Each such server contains identical information.) The registration server maintains a data base that associates with each registered SN some authentication information such as a password or encryption key, and a list of network addresses. The network addresses locate *message servers* (discussed in section 4.1) willing to buffer messages for that recipient.

The protocols supported by a registration server are:

> *registration* - a variety of protocols to allow maintenance of a registration server's data base, and communication of changes to all instances of registration servers for the same naming authority;
>
> *authentication* - given an R-name and an authenticator, return the result *good* or *bad*, depending on whether the authenticator matches the one associated with the R-name in the data base;

location - given an R-name, return the associated list of message server addresses.

3.2 Finding registration servers

In order to perform the location protocol mentioned above, it must be possible to map a naming authority name into a network address of one of the associated registration servers. For this mapping function we rely on existing *name lookup servers* in the network. Each contains a copy of a data base that maps character strings into lists of network addresses. Any computer attached to the network can find a name lookup server and obtain the network addresses associated with a given character string. (A name lookup server is located by broadcasting a packet on the directly connected Ethernet local network.) We include in the name lookup server data bases the names of all naming authorities, and associate with each the list of network addresses for the corresponding registration servers. Since the set of naming authorities is much smaller and changes much less often than the set of all R-names, the existing centralized, manual procedures for maintaining the name lookup server data base are adequate.

Figure 2 illustrates the complete protocol for locating a message server that buffers a recipient's messages, given the recipient's R-name.

3.3 Distribution lists

It is important that a message system support delivery of messages addressed to groups as well as individuals. Distribution lists, named sets of recipient names, are a common way to provide this function. We provide distribution lists by allowing an R-name to identify either an individual or a distribution list. If an R-name names a distribution list then the registration server's data base contains the set of constituent R-names and the R-name of a maintainer for the distribution list.

When a registration server is requested to create a new distribution list, the requestor supplies the R-names it is to contain. (They need not all be in the same naming authority.) The requestor's R-name is recorded as the maintainer. Thereafter, requests to change the content of that distribution list, or delete it, can be made only by the maintainer. The authentication protocol of the maintainer's naming authority is used to authenticate the identity of someone claiming to be the maintainer.

With the addition of distribution lists, the location protocol mentioned above must be extended to return the set of constituent R-names when the R-name being located names a distribution list. Also, the set of registration protocols must be extended to allow creation, maintenance, and deletion of distribution lists, as well as communication of distribution list changes to multiple instances of a naming authority's registration servers.

4. Delivery

The primary function of the transport mechanism is to deliver messages from a sender to a set of recipients. On the surface this appears to be straightforward and uninteresting, but when coupled with the reliability goals of section 2.1, delivery can become quite complex. We first present a "normal case" scenario illustrating the distribution of function, then consider the complications that can arise from errors at various stages.

4.1 A Message Delivery Scenario

A user has prepared a message for delivery to a list of recipients. His interactive message manager (the client) maps the recipient list into a list of R-names, then establishes a connection with any conveniently accessible message server. The server asks the client for the user's R-name and authenticator and then uses a registration server for the user's naming authority to check that the user is legitimate. Following successful authentication, the message server obtains the list of recipients from the client and, using appropriate registration servers, checks that all recipients are registered message system users. The client then supplies the message text to be transmitted and terminates the connection.

The message server now has the responsibility of delivering the message to each recipient or returning it to the sender. It again uses the appropriate registration servers to locate each R-name in the recipient list. (The discussion of caching in section 6 shows why this second logical reference to the registration server for each recipient is reasonable.) If the R-name corresponds to an individual, the server provides the name of a message server that contains the *mailbox* for that individual. (In reality, the registration server provides an ordered list of message servers. The use of this list is discussed below; for now we assume it has only one element.) If the R-name corresponds to a distribution list, the registration server provides the list of R-names that comprise that list. The message server then maps each of these names in a similar fashion. When the original recipient list has been completely expanded and duplicate R-names eliminated, the transport mechanism has produced a list of pairs <RN,MS>, where RN is an R-name of a recipient and MS is the message server containing RN's mailbox.

The originating message server must now deliver the message text to the destination servers. It first sorts the <RN,MS> pairs by MS and then sends each distinct MS a single copy of the message text with a list of RNs that have mailboxes at that MS site. Each receiving mail server is then responsible for storing the message text and making it available when a client acting on behalf of any of the indicated recipients subsequently requests it.

4.2 Reliable Delivery

There are many opportunities for error in the preceding scenario. Indeed, the delivery algorithm is so obvious that the only interesting aspect is its response to exceptional conditions. Let us reconsider the steps of the delivery process, asking at each point: "What can go wrong here?"

What if some of the R-names supplied by the client are invalid? The originating message server supplies the client with a complete list of invalid R-names. The client can then prompt the user for correction of all erroneous recipients. (Recall that distribution list expansion has not occurred yet, so invalid constituent R-names are not reported to the sender.) After correction of the recipient list, the sender can resubmit the message for delivery.

What if, after accepting a message, the transport mechanism finds an erroneous R-name? If the name appeared in the original recipient list, then the intended recipient must recently have become unregistered. The sender should be notified by returning the message to him with an appropriate explanation. If the name came from a distribution list, the transport mechanism should notify the maintainer of the distribution list (whose name is stored with the distribution list) and optionally the sender of the message. The nature of these notifications is quite different, since the sender will not necessarily recognize the invalid R-name but the distribution list maintainer will (or should). There are a number of low-probability, second-order error cases to consider. What if the list maintainer's R-name is invalid? What if the attempt to return to sender fails because the sender's R-name is now invalid? There is probably no single "right" solution to these situations, but as long as the transport mechanism does something reasonable (like logging the problem in a known file for independent, human examination), any solution will do.

What if a destination message server receives a message for recipients whose mailboxes it doesn't contain? In this situation it should return the message to the originating message server with a list of the R-names whose mailboxes are not local.

What does the originating server do if a destination server is unavailable or rejects a message? As noted above, each R-name actually has an ordered list of message servers to which messages may be sent. The originating server simply tries each one in turn until one accepts the message. (This implies that a receiving client must check the mailbox in each server in its user's list. In practice, we expect that lists of length two will be the normal case.) Of course, if no server in the list is willing (or able) to accept the message, the situation is more serious. If the problem is that none of the destination servers are operational, the message is queued by the originating server and delivery retried later. However, if all destination servers deliberately reject the message, then the transport mechanism may have detected an internal inconsistency and the message is returned to the sender (if possible) with an explanation (and apology).

Few recovery strategies work all the time. In fact, the practical goal of reliable system design is to reduce the probability of failure to an acceptably small value. The registration server data bases change slowly (over hours, not milliseconds) with respect to message transmission time, which simplifies the recovery algorithms considerably. Multiple failures within a short time interval are unlikely and can be accommodated with one or two catch-all algorithms. In fact, as long as a sufficiently general "recovery state" is maintained with the message as it moves around the network, recovery actions may be easily extended over time as the predominant failure modes are identified.

4.3 Guaranteed Properties of Messages

A message system should be able to guarantee the accuracy of certain information associated with each message it transmits. Users will not trust the message system unless they are certain that every message they receive was actually sent by the apparent sender. They may want guarantees of the accuracy of other message properties as well, e.g., the list of recipients. Because the client programs employed by various users may not all be identical and may be subject to tampering or corruption in a particular personal computer, the burden of guaranteeing these message properties falls on the transport mechanism.

Typically, the information a recipient wants to have guaranteed appears in the *internal header* of a text message, a human-readable prologue that indicates the message's sender, recipients, time of delivery, and other similar information. This header is part of the message body, and thus is constructed exclusively by the sending client. The transport mechanism regards the entire message content as uninterpreted data, and consequently cannot vouch for the accuracy of the information in the internal header. Unless the transport mechanism provides an independent means of verifying the header information, a malicious (or malfunctioning) sending client could falsify information in the internal header of a message and thereby mislead the recipients.

Information whose accuracy the transport mechanism guarantees will be stored in a *property list* associated with the message. The property list serves two distinct functions: it holds information required by the transport mechanism to route the message from sender to recipients, and it records properties of the transaction with the sender that may be of interest to the recipients. The latter category includes the length of the message text, the (authenticated) identity of the sender, the complete list of recipients, and the time the message was presented for delivery. The property list is constructed by the transport mechanism during the sending transaction with the client and cannot be directly modified by the sender. Consequently, the recipients are assured that the property list accurately describes the origin and destination of the message. We assume that the transport mechanism is less susceptible to corruption than individual clients.

We believe that a practical message system must provide at least the guarantees described above. However, we can imagine extending the property list mechanism to provide other guarantees. For example, we could easily use it to implement a "return recipient" feature, which notifies the original sender when the recipient receives the message. By "receives" we mean that the receiving client retrieves the message from the mailbox in which the transport mechanism stored it. To delay the return receipt until the message is read by the user requires the cooperation of the (possibly untrustworthy) receiving client.

5. Implementation Notes

We have not finished building the transport mechanism described in this paper. Discussion of several details of the intended implementation, however, will help to substantiate and validate the design presented above.

We plan to attach registration server and message server computers to the network at various places. (A registration and a message server may cohabit the same machine.) The message servers will use their local disks as first choice storage for mailboxes. While most recipients will retrieve newly arrived mail from their mailboxes within hours of its arrival, some recipients will occasionally go for days or weeks without emptying their mailboxes. When the local disk becomes clogged with unretrieved mail, the message server will move old mail to existing file servers. The use of file servers by the message servers will be transparent to message server clients. Emptying a mailbox that includes old messages actually stored on a file server will cause the message server to first move the old messages back to its local disk.

Caching of registration and location information in the message server machines is an important technique for producing an efficient, responsive transport mechanism. We expect that each message server will maintain a cache of the last several hundred <R-name, mailbox location list> pairs. Since this cache will be shared among all users of a particular message server to deliver outgoing messages, we expect it to contain the names of most recipients for the local area served. Thus, location and validation will normally avoid interaction with the registration server and can be quite fast. The advantage of a shared cache is a primary argument in favor of doing recipient validation and location in the message server rather than in the sending client.

The use of a cache, of course, generates some problems of its own, the most troublesome being that cache entries become invalid as the registration server data bases change. We plan to take advantage of the error messages returned from destination message servers to the sending message server to trigger invalidation of cache entries and replacement with valid data from the registration server. In this way the registration servers are relieved of any responsibility for informing message servers that registration data has changed.

We have not decided yet if the content of recently used distribution lists will be cached.

6. Internetwork Forwarding

The naming mechanisms discussed in section 3 work well when the transport mechanism can impose a single, network-wide name interpretation. However, special measures must be taken when connections to other networks exist. Since such networks impose their own naming requirements (which may be different or incompatible), some translation of names is inevitable.

Consider, for example, a connection to the ArpaNet [ArpaNet]. More precisely, assume that some server has a physical connection both to the network described in section 1.1 and to the ArpaNet. Thus, this server is, at least in some respects, an ArpaNet host. ArpaNet message recipient identifications typically take the form: "UserName@HostName" and thus cannot be directly interpreted by the normal conventions discussed earlier. We can accommodate them quite easily, however, as follows:

1) We identify ArpaNet recipients by the construct: "ArpaNet.<ArpaNet user identification>". Thus "X@Y" becomes "ArpaNet.X@Y". We establish "ArpaNet" as a naming authority (in the sense of section 3.2) and use the server possessing the ArpaNet connection as the registration server and message server for that naming authority.

2) When we attempt to send a message to "ArpaNet.X@Y", the transport mechanism communicates with the ArpaNet registration server to determine the validity of, and mailbox site for, "X@Y". Since the ArpaNet does not support network-wide registration of users in a fashion that permits dynamic validation, our internetwork message server assumes that "X@Y" is a valid user and identifies itself as the site of Y's mailbox. When the message eventually arrives at the internetwork message server, this server changes the form of each R-name in the internal header. For R-names in the "ArpaNet" naming authority, "ArpaNet." is removed. For all other R-names "@<HostName>" is appended, where <HostName> is the name by which the internetwork message server is identified as an ArpaNet host. This server then forwards the message to the ArpaNet using the required protocol, acting as a client of the ArpaNet transport mechanism [Kalba].

3) Incoming messages are handled by the obvious inverse algorithm. Then the message is treated as though it originated at a local client.

This remapping of names via the naming authority mechanism appears quite flexible for handling naming incompatibilities in interconnected networks. The only apparent difficulties lie in validating recipients and authenticating senders when the necessary on-line facilities are not available in the foreign net. The simplest approach to recipient validation, as illustrated above, is to assume recipient names are valid, and report subsequent failures in delivery by returning the message to the sending network with an appropriate explanation. There is no simple way to authenticate the senders of a message arriving from a foreign network.

7. Final Remark

The distributed transport mechanism described in general in this paper is now under construction. We plan to report insights gained from detailed design, construction, and use of the system in future papers.

Acknowledgements

Colleagues who have participated in the design of the transport mechanism described in this paper or offered helpful advice are Roger Needham, Ed Taft, Ben Wegbreit, Andrew Birrell, Jim Horning, Doug Brotz, and Dave Boggs.

References

[ArpaNet] Feinler, E. and Postel, J., eds., *ARPANET Protocol Handbook*, NIC 7104, Network Information Center, SRI International, Menlo Park, Ca., January 1978.

[Kalba] Kalba, Konrad K. et. al., *Electronic Message Systems: The Technological, Market and Regulatory Prospects*, Kalba Bowen Associates, Cambridge, Mass., April, 1978.

[Metcalfe and Boggs] Metcalfe, R.M. and Boggs, D.R., "Ethernet: Distributed Packet Switching for Local Computer Networks", *Communications of the ACM 19*, 7 (July 1976), pp. 395-404.

[Needham and Schroeder] Needham, R.M. and Schroeder, M.D., "Using Encryption for Authentication in Large Networks of Computers", *Communications of the ACM 21*, 12 (Dec. 1978), pp. 993-999.

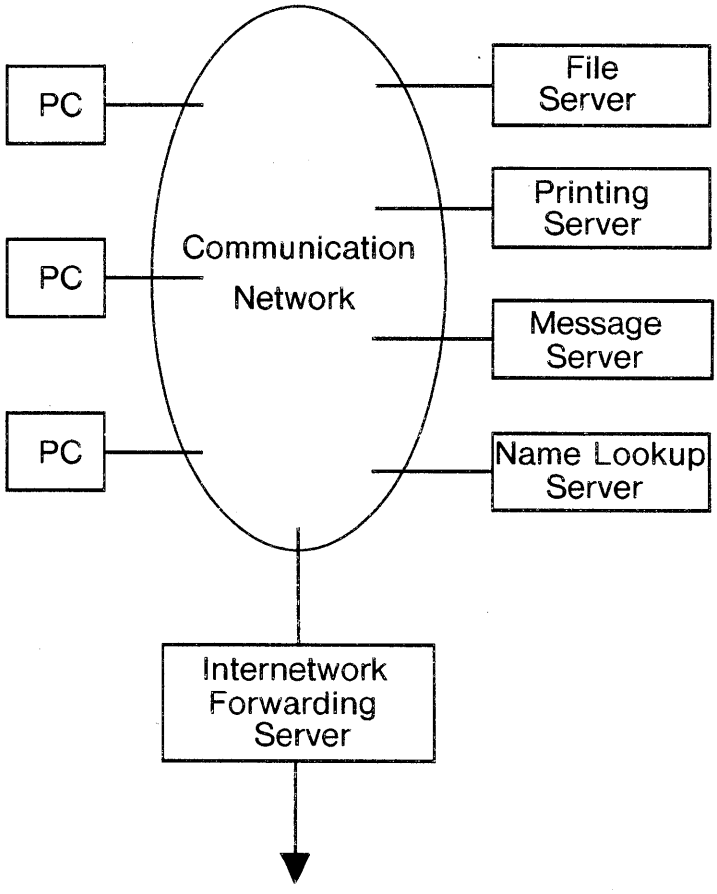

Figure 1. Components of the Xerox Network

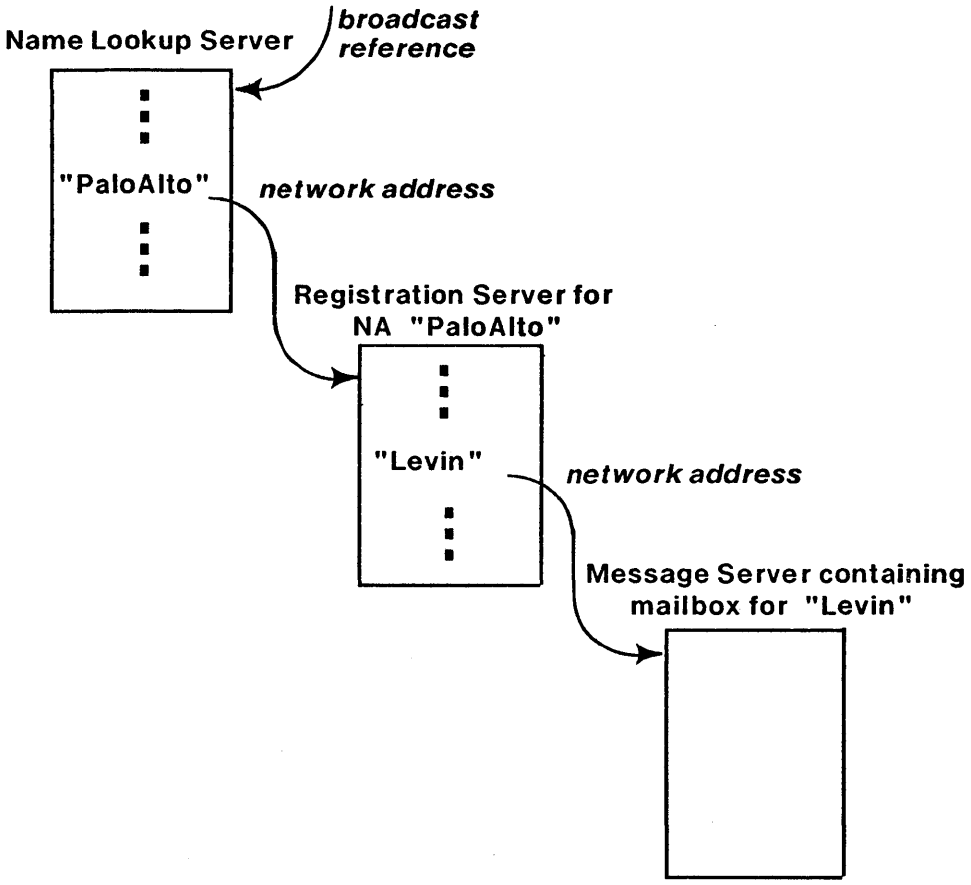

Figure 2. Location Protocol for the Name: "PaloAlto.Levin"

XEROX

Transport of Electronic Messages Through a Network

By Roy Levin and Michael D. Schroeder

CSL-79-4

XEROX

Xerox Corporation
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304