MIT/LCS/TM-263

NAMING AND DIRECTORY ISSUES IN MESSAGE TRANSFER SYSTEMS

Marvin A. Sirbu, Jr.

Juliet B. Sutherland

July 1984

Naming and Directory Issues in Message Transfer Systems

Marvin A. Sirbu, Jr.
Juliet B. Sutherland

Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts
U.S.A.

ABSTRACT

A message transfer system requires some means for users to determine the addresses of their correspondents. A *Directory Service* aids users in identifying a particular correspondent and the correspondent's address. In this paper we discuss the technical, economic, organizational and political requirements which must be satisfied by a directory service. We develop a language for describing alternative architectures for directory service borrowed from notions of hierarchical computer file system design. We propose a system of naming and directory services which meets the stated requirements based on names which specify a *path* through a sequence of directories. Finally, we compare our proposal to several alternative designs for directory service which have appeared in the literature.

This paper appeared in the proceedings of the IFIP Working Group 6.5 International Working Conference on Computer Message Systems, May, 1984, Nottingham, England.

This work was supported in part by the Advanced Research Project Agency of the Department of Defense, monitored by the Office of Naval Research under contract N00014-83-K-0125

Table of Contents

List of Figures

1. Introduction

Technologies for sending mail have evolved considerably since early writers used clay tablets, papyrus or parchment for recording messages which took months to deliver by hand. Computer mail offers the possibility of near instantaneous delivery of messages in electronic form. A common problem in message systems, whatever the form, is specifying the recipient and his or her address. When there was no reliable postal service, messages were sent with anyone going in the correct direction and addresses consisted of an approximate description of where to find the person. One ancient Egyptian letter carrier, having arrived in the correct town, had to cope with the following:

> From Moon Gate [i.e., the particular gate of several in the town's defense wall that he was to enter by], walk as if toward the granaries ... and at the first street back of the baths turn left.... Then go west. Then go down the steps and up the other steps and turn right. After the temple precinct there is a seven-story house with a basket-weaving establishment. Inquire there or from the concierge.... Then give a shout. [2]

One wonders what would have happened if the letter writer had not known the town well enough to give directions.

As reliable postal systems became common and addresses became standardized, the problem shifted from providing a description of how to find the person to determining the person's standard address. Some towns published directories listing the streets and house numbers of residents. However, the generalized problem of finding postal addresses continues to this day. A similar problem was created in 1876 with the invention of the telephone. Prior to automatic switching, there was no standardized "address"—*e.g.* telephone number—for subscribers; nevertheless, written subscriber directories first appeared in 1878 [21]. In the mid 1960s, following the introduction in 1964 of the first convenience facsimile machines, Xerox published a directory of facsimile users. The practice was quickly abandoned when Xerox discovered that its competitors were using the directory as a list of prospects.

Although the directory problem of determining addresses exists in all communication systems, at present only the telephone system has explicit, widely available, printed directories and a service for obtaining telephone numbers. In the future, we believe that a worldwide electronic mail system, like the telephone system, will find some form of directory service indispensable.

Addressing and directory issues are just beginning to be addressed in world standards organizations such as the CCITT [14]. We have not seen, however, a comprehensive discussion of the many issues raised in the provision of directory services for electronic mail. Our purpose in this paper is to provide a starting point for such a discussion. To this end, we set forth some goals for a worldwide directory system, discuss naming issues as they pertain to directories, and present our

views on a selection of architectural and usage issues. To provide some perspective, we conclude by examining several existing or proposed directory systems to see how they have addressed the issues we have raised.

2. Goals

The design of a directory system is complicated by the need to meet a number of different, and at times conflicting, goals. In order to facilitate design, and the comparison of alternative designs, we have identified the following goals or criteria for evaluating a proposed directory system and service.

1. *The directory service should be easy to use.* It should provide a simple means for users to find the addresses of their correspondents.

2. *The design must be scalable.* We are proposing a design for a world-wide directory service which should eventually grow to accommodate literally billions of correspondents. The design should allow for efficient operation at that scale. At the same time, it should be implementable at reasonable cost on a smaller scale so that getting started is not a large hurdle.

3. *The service should provide for a high degree of automation.* Economic operation at a scale of billions of users requires a high degree of automation.

4. *A directory service should always be available.* To insure that directory service is always available, the design should allow for replication of information and of service providers.

5. *Directory information should be correct and up to date.* A good design should make it easy to update directory information. It should also provide procedures which insure that the information is accurate.

6. *Personal privacy must be protected.* A directory is a database of information about an individual. As with all such databases, it must provide for the protection of individual privacy.

7. *Corporate or organizational privacy must be protected.* Organizations may also have privacy concerns separate from those of individuals. These concerns must also be respected in the design of a directory service.

8. *The design must provide for multiple unrelated organizations to both compete and cooperate in the provision of directory service.* This requirement expresses our belief that directory service is a business in which numerous unrelated organizations will engage, often in competition with each other. A directory service design must be capable of operation without assuming a level of cooperation among participating entities beyond that which can be expected in a competitive marketplace.

We are certain that other criteria could easily be added to this list. One of our goals in this paper is to begin to set down explicitly the design criteria behind alternative approaches to the directory service problem.

3. The IFIP Model of a Message Transfer System

Much recent work on electronic mail has adopted a model[1] first set forth by IFIP Working Group 6.5. [5, 14] In this model, the major components of a message system are the User Agent (UA) and the Message Transfer Agent (MTA). A user, known as an originator or a recipient depending on his or her role in the communication, sends or receives mail through the User Agent. A UA might provide a text editor, a file system, or other tools for preparing and storing messages. The Message Transfer System (MTS) is an interconnected network of UAs and MTAs which moves messages from one UA to another. A UA submits a message to an MTA, which either delivers it to the destination UA or relays it to another MTA nearer the final destination. All of these concepts are illustrated in Figure 3-1 which is adapted from Redell and White [14].

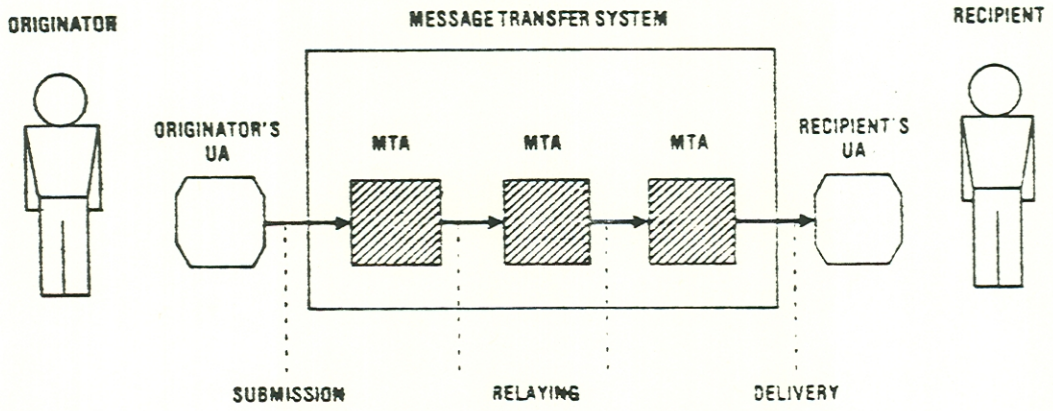

Figure 3-1: A model of an electronic mail system.

UA's and MTA's can be grouped together in any combination to form a Management Domain (MD). A management domain is an organizational unit that is distinct from the physical architecture, that is, the interconnection of MTAs and UAs, of the system. Within a management domain, MTA's and UA's may use protocols different from those prescribed in a message transfer standard. At the boundaries between all management domains, however, the standard protocol will be employed.

The conceptual model just described can also be expressed in terms of layers following the principles of Open Systems Interconnection [9]. The message system model has two layers, the User Agent Layer (UAL) encompassing the functions of the UA's, and the Message Transfer Layer (MTL), which contains the functionality of the MTS. These layers are shown in Figure 3-2. The layered model defines three entities within the two layers. The UA entity in the UAL and the MTA entity in the MTL have already been described. The Submission and Delivery Entity (SDE) is present only if there

[1] The model described here is actually the IFIP model as adopted and changed by the CCITT and described in [3, 4].

is no MTA entity colocated at the same site as the UA entity. The SDE does not provide transfer services but does provide the interface between the UA and the MTA. These entities are shown in Figure 3-2.

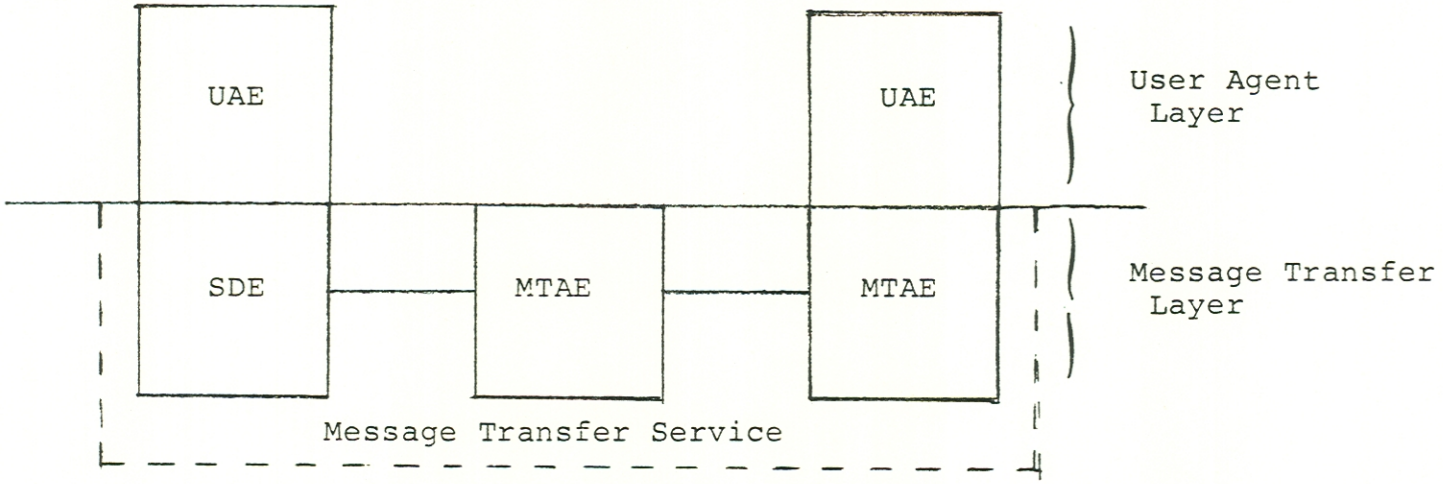

Figure 3-2: Alternate model showing layers, entities, and their relationships.

IFIP Working Group 6.5 has also provided some terminology for directories [8]. Directory services are provided by one or more *Directory Service Agents* (DSAs). A DSA can communicate with either a UA or an MTA and is outside of the UAL or MTL layers. The *client* is the entity which is requesting information from the DSA. We define further the *target* of the inquiry as the person or service about which information is being requested.

4. Naming and Binding

What's in a name? A rose by any other name would smell as sweet...

—William Shakespeare

Any discussion of a Directory Service must first begin by clarifying the meaning of the various terms to be used in the discussion, and most particularly the concept of a *name*.

IFIP Working Group 6.5 has described a *name* as

> "...a linguistic object that corresponds to a particular entity in some universe of discourse germane to the language in which the name is expressed. The correspondence between names (in the language) and entities (in the universe of discourse) is the relation of denoting. A name denotes or identifies the entity with which it is paired." [8]

Another view of naming is given by Saltzer who describes naming as the mapping or binding of a higher-level semantic construct into a lower-level construct. [16] This lower-level construct may be viewed as a name for a still lower level construct until the most fundamental physical constructs are

identified.

For example, several mappings or bindings are required to access a service over a computer network: [17]

- Binding of a service to a computer node from which the service is available;
- Binding of a computer node to an attachment point on a computer network;
- Binding of an attachment point on a computer network to a route from the source to the attachment point.

A *context*, according to Saltzer is, abstractly, a particular set of bindings. A *name* is a character or bit-string identifier which is bound to a lower level construct in some context.

Key to Saltzer's notion of names, is that they do not exist apart from a "context" or abstract mechanism within which they are resolved. This "context" may be a piece of hardware—as in the computer circuits which map the "name" of a piece of data (its hardware address) to a particular data item in memory. More often, the context is a data table in which the name appears along with the lower level name to which it is bound.

The city at the geographic location 51° 32' N, 0° 5' E. is *named* London in the context of an English language Atlas. In a French Atlas, that location identifier is associated with the character string "Londres". Each name is valid only in the context of a particular Atlas.

More concretely, a *directory* is an object consisting of a table of bindings between names and objects; a directory is an example of a context. A *directory service* takes a *name* as input, and as output produces the lower level object to which the name is bound. For example, given the "name" of the user of a message handling system, a directory service might return the "name" of the User Agent to which mail for that person should be addressed. The entries in the table are the *names*, or identifiers, of instances of each type of object or construct of interest.

As with layering in protocol design, the purpose of multiple levels of binding is to allow higher levels to be oblivious to changes in lower levels. A binding allows users of higher level semantic constructs to use an unvarying name to refer to a construct, while allowing for changes in the lower level constructs to which the first construct is bound. Thus, one might want to specify that some mail is to be sent to the complaint department at the local department store without being concerned that the construct identified by "complaint department" might be mapped to different User Agents at different times.

Multiple levels of binding also facilitate sharing of information that may be used by more than one service. Thus, a binding of a host to a network attachment point may be used in the process of mapping a variety of different services to their network addresses.

One goal, then, in developing a system of names and bindings for use in electronic mail networks, is to allow parts of the system to use unvarying names while allowing the binding of those names to other, lower-level entities, to change.

If we examine the IFIP model for a message handling system, we can identify a number of entities that require names and bindings. These entities include:

1. *Users*
   Users are outside the message transfer system. The MTS serves the needs of users. A variety of different entities might be users of an MTS:

   a. *Person*
      A person is a particular human being. The human being may change jobs, change employer, change residence, even change names, but is still the same person. Frequently we wish to send a communication to a uniquely identified human being.

   b. *Roles*
      A role is a job title, or functional responsibility within an organization or social structure. Sales manager, affirmative action officer, parent or guardian, and mayor are examples of generic roles. A specific role generally requires further qualification such as "sales manager for Massachusetts for the ABC line of products from the XYZ corporation". Frequently we wish to address a communication to the person (or computer service) currently acting in a particular role. Roles are bound to one or more persons and the binding may change occasionally or at regular intervals (*e.g.* shift supervisor). Roles are created, altered and destroyed at intervals ranging from months to years. There is typically a many to many binding of Persons to Roles.

   c. *Personnae*
      Most working persons will receive mail — as they do today — at both the home and the office. We might view these as separate roles, but roles which are always bound to the same person. We define *personnae* as a set of roles which — though they may be created and destroyed — can never be bound to any other person. Thus "Marvin Sirbu at work" and "Marvin Sirbu at home" are different personnae. These personnae will generally correspond to different User Agents — one at the office paid for by the employer, and one at home, paid for by the household.

d. *Computer Service*
The originators or recipients of electronic mail may be computer services as well as persons. Thus, computer services may appear as entities in a directory service.

2. *Distribution List*
A Distribution List is a protocol entity which consists of a list of several users.

3. *User Agent*
A User Agent is a protocol entity which provides services to an individual user in reading and sending mail. A User Agent may be bound to a Personna, to a Role, or to a Computer Service.

4. *Message Transfer Agent*
Message Transfer Agents accept messages from and deliver messages to User Agents.

5. *Management Domain*
A management domain is a portion of the total Message System that is controlled by a particular organization or group of cooperating organizations. It may contain numerous Message Transfer Agents and User Agents. A management domain may or may not control other parts of the Message System.

This partial list of entities in a Message Transfer System illustrates the types of objects for which bindings might be found in a directory. As suggested above, a directory is a table of bindings from symbolic names to objects or other symbolic names.

In the context of a Message Transfer System there are at least three bindings of interest which should be provided by a directory service. [8] Users approaching a Message Transfer System may have in mind a particular person, a human user of the MTS, to whom they wish to direct mail. They may know some attributes about the person, such as his or her surname, given names, place of employment or home address, and whether they wish to reach the person at home or at work. A first level of binding provided by a directory service takes from a client a set of attributes of a personna and maps them into a particular form of identifier for that personna, which we shall call a PersonnaName.[2] A second binding maps a PersonnaName—the name of an object outside the Message Transfer Service, but necessarily part of the Directory Service—into the name of a User Agent, an object inside the MTS. Finally, a third binding maps a User Agent Name into a User Agent Address. We leave to section 4.5 a discussion of the latter two bindings, as well as other bindings which might be necessary in an MTS.

---

[2]Similar arguments apply if the destination is a role or a service as opposed to a personna.

4.1. Names in a Message Transfer System

A PersonnaName is clearly a central object in a directory. What characteristics would we like this PersonnaName to possess? It is helpful to consider first some general choices in designing names, and then the specific requirements for a PersonnaName.

Human versus computational use. [16]

Names intended for use by human beings should consist of character strings as opposed to bit strings or numeric strings. They should be mnemonically useful. Ambiguity may be acceptable, if it can be resolved through interaction with a human being. In general, such names should be selected by human beings, not assigned by machines.

Names for computational use need not have mnemonic value. However, they must be absolutely unambiguous, so that they can be resolved by machines without human intervention.

Flat versus structured names.

Names can consist of arbitrary strings which provide no additional information. Or they can be structured from multiple components which can permit the directory to be logically or physically partitioned according to the components. This partitioning may be desirable for reasons of scalability, efficiency, or privacy (Cf Section 4.2).

Component values can be assigned by separate *naming authorities*. This allows delegation of naming to distributed authorities while assuring that all names are universally unique.

Characteristics of PersonnaNames

Of the general considerations described above, which are particularly applicable to PersonnaNames in a directory service?

1. A PersonnaName should uniquely identify an individual personna, so that a machine could automatically map a PersonnaName into a User Agent Name without further interaction with a human user.

2. The PersonnaName should not change even though various other attributes of the person, such as address, employer, or marital status should change. Thus, once we have found someone's PersonnaName, we can trust that that name will continue to identify the same person when we use the directory to find the current User Agent Name bound to that PersonnaName.

3. The PersonnaName should be a multi-component name which allows the directory to be

logically or physically partitioned for reasons of efficiency, privacy, and decentralized naming authority.

4. If the PersonnaName could be easily remembered or guessed by human beings, users would not need to utilize the directory service to find the PersonnaName for a particular user.

In the real world, it is virtually impossible to meet all of these criteria simultaneously. Criteria 4 might suggest the use of Given Names plus Surnames as PersonnaNames. However, Given Names + Surnames do not, in general, uniquely identify an individual, violating requirement 1. Women often change their surnames when they marry, violating requirement 2. In addition, Given Names + Surnames do not provide for a natural partitioning of the name space among directories as specified in point 3.

The concatenation of {Surname} + {Given Name} + {Address} + {City} + {State} + {Residence|Office} is used by most telephone companies to provide a unique PersonnaName for deriving a telephone number from the telephone directory service. The problem with this approach is that since the binding between a Person and a StreetAddress changes over time, it is as if the unique identity of the individual were to change whenever he or she moved. Other solutions, involving "attributes" of a person such as Employer or Department also suffer from the same problem. Each of these attributes represents a (possibly temporary) binding. If the binding should change, we would have to update our tables showing the binding between Person and User Agent, not because that binding had changed, but because the way we identify the Person entry in the table had changed. Indeed it would make more sense to use an unvarying attribute, such as someone's city of birth, as opposed to their current residence, for constructing a unique PersonnaName.[3]

The concatenation of {Country} + {National ID Number} would provide a unique name, but would fail to meet requirement 4. It would meet requirement 3 only if the ID number were itself structured into components, and even then, the partitioning of the name space implied by the use of ID number components is not natural in terms of directory management.[4]

Note that we do not require that a personna have only one PersonnaName. There can be many PersonnaNames which will reference the same User Agent Name, and thus can be considered

---

[3] As evidence that human beings have long since discovered this principle, we have only to note that a sentence such as "John of York is living in Kent" makes perfect sense.

[4] AT&T's *Expanded 800 Service* allows permanent "personal" phone numbers to be mapped into any physical number on the network. The distributed database is partitioned by the exchange code. [7]

synonyms of each other for the purpose of binding PersonnaNames to User Agents. In an environment in which multiple organizations will likely maintain directory services partitioned along different lines, we can expect clients to employ a variety of PersonnaNames, resolved through a variety of directory services, in order to map users into User Agents.

In practice, therefore, we have little choice but to use as PersonnaNames, names which depend on several attributes of the individual, even if these represent bindings subject to change. The key is to use attributes which change less frequently than the binding we are trying to determine.[5] By using a multiplicity of attributes, and particularly attributes which do not change (place of birth; graduated from MIT; etc.) we can improve our ability to reliably identify an instance of a Person.

4.2. Names and Partitions

In the simple case of a small network or community of users, it is feasible for a single DSA to store the entire database of users. We can expect, however, that the community of interconnected users—including persons, roles, and services—will eventually grow to be of the order of the number of telephones in the world, or even larger -- *e.g.* several billion users. Given our desire for reliable and economic operation, protection of privacy, and efficient management of updates, maintaining a single database of that size—even if service is provided through multiple servers each with copies of the database—is not a feasible option.

We can suppose, therefore, that there are multiple DSAs each with a fraction of the database. What might be strategies for partitioning the name space among DSA's? There are several principles of division.

1. *Geographical* — The world telephone network partitions directory responsibility into geographic regions generally corresponding to recognized civil boundaries.

2. *By Management Domain* — Each management domain will probably maintain a list of names of its users. In fact, if the management domain and naming authority are the same, the management domain must maintain such a list.

3. *Organizational* — DSA's may be maintained by organizations independently of whether they constitute a management domain. Thus the IEEE maintains and publishes a directory of its members, although it is not likely to be the provider of mail service to them.

---

[5] We would not want to identify John Brown as the person who gets his mail at JBROWN@MC in a table binding Persons to street addresses!

4. *Hybrid* Various combinations of the above divisions can also be imagined.

Indeed, it is likely that individual names will appear in multiple DSAs with overlapping partitions of the entire name space. In order to assure availability of the directory, it is also necessary that any given partition will be maintained on more than one service machine.

In order to take advantage of a partitioned directory, it must be possible by inspecting a name to determine in which directory to look. This implies that names are not arbitrary strings, but in fact have some structure, or components, which provide information about the partition structure of the database. There are two quite distinct alternatives for conceptualizing a multi-component name.

A "tuple-name" is a multi-component name which is regarded as a "tuple" in a relational database. Each component of the name corresponds to a particular field type of a *name-address record* (Figure 4-1).

⟨country⟩	⟨organization⟩	⟨personal name⟩	⟨org. discrim⟩	. . .	⟨UA⟩
USA	MIT	Marvin Sirbu	Lab for Comp Sci	. . .	⟨UA⟩
UK	Univ. of London	John Smith	Dept of Comp Sci	. . .	⟨UA⟩

Figure 4-1: Names as Tuples in a Relational Database

To use the directory service a client must supply sufficient field values to identify a unique tuple. When that tuple is retrieved from the database, the address field (User Agent Name) of the tuple is also retrieved.

The "tuple-name" requires that each field which might appear in the tuple be well defined. Multiple directories would presumably use the same record structure, i.e. the same fields, for the tuples in their portion of the name-address database. Having names for fields would allow the client to present components to the Directory Service in an unordered fashion; named fields would also facilitate guessing.[6]

---

[6] Personnal communication from Debra Deutsch.

An alternative view is to regard a multi-component name as a *path name*. Each component of a path name is an entry in a directory. Opposite the entry is either a User Agent Name, or the address of another directory in which the next component of the path name is resolved (Figure 4-2). This view of names is similar to that used in various computer filing systems. [16].

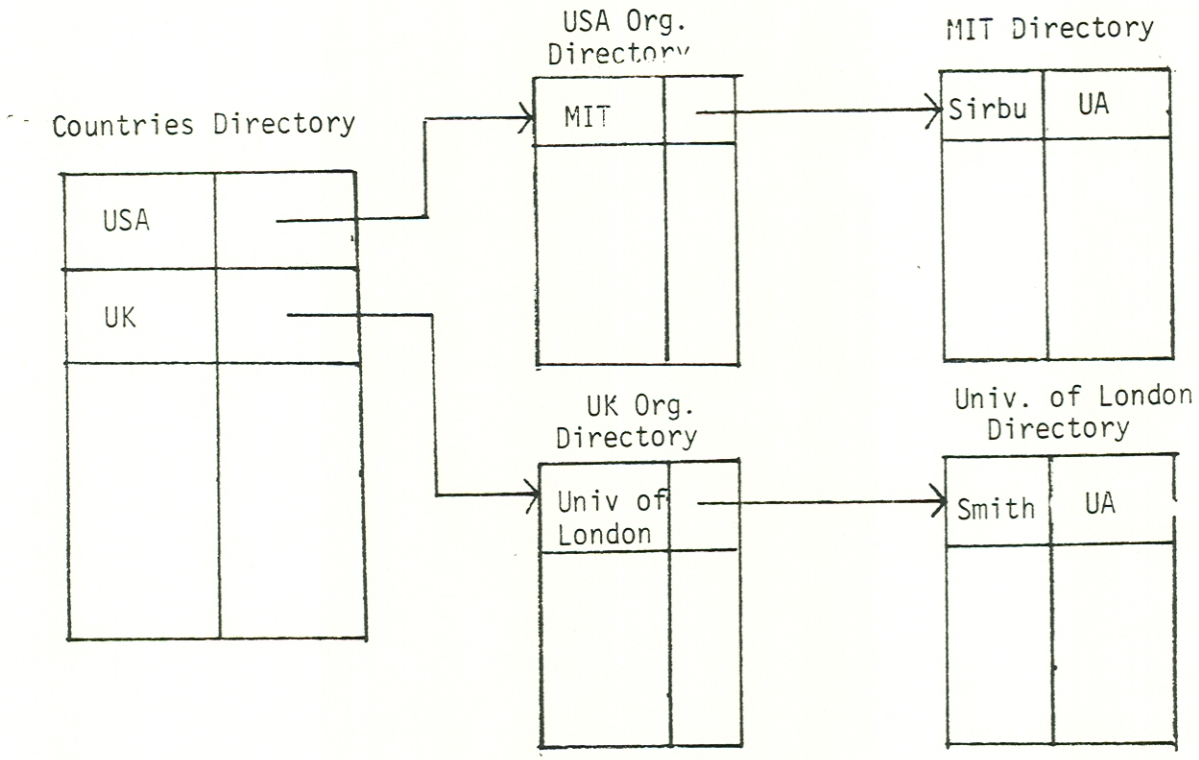

Figure 4-2: Multi-Component Names as Path Names

With path names, there is no *a priori* expectation that the component entries represent some "field" type with specific characteristics. A component simply represents an entry in *some* directory. It is reasonable to assume that the entries in a particular directory might have something in common — for example, the ITU might maintain a directory whose entries were official names of countries, and which pointed to national directories in each country.

However, such a meaning is not required. For example, one entry in the "Stanford" directory may be the component "MIT" and a pointer to the "MIT" directory. Thus persons at Stanford could query the local Stanford directory to resolve the path name (STANFORD)MIT.JULIET SUTHERLAND.[7]

---

[7] The notation *(<first directory>)<name1>.<name2>...<nameN>* should be read: the first component, *<name1>* is resolved in directory *<first directory>*; subsequent components *<namei>* are each resolved in the directory pointed to by the previous component.

As the above example makes clear, a path name can be resolved only if one knows the directory in which the first component of the path name is to be found. That is, a path name is not complete without specifying an initial directory. Traditionally, file systems use two approaches to resolving this problem. The first approach assumes the name is relevant to some default working directory, which may be different for each user. The second assumes that directories are organized not into an arbitrary *naming network*[8] but as a *naming hierarchy* or rooted tree. [16] The first component of the path name is always resolved in the root directory. The root directory itself can be found in one of two ways: either every directory contains a pointer back to the root directory; or, every directory contains a pointer to its parent in the hierarchy, and this chain can be traced back to find the root. [15] Once the root directory has been located, the path name — which for this specific case might also be called a *tree name* — is resolved starting with the root. A third alternative, used in the Clearinghouse design, assumes that there is a group of directories at the "top" level, all of which have pointers to each other. Thus, starting with any one of these "top level" directories it is possible to resolve any name.

For a universal MTS directory service, the use of tree names could require every name resolution to examine the root directory, which is clearly inefficient. An alternative, therefore, is used in the ARPA Domain Name proposals to reduce the load on the root directory. In this name/directory design, a directory may contain a multi-component entry which allows some directories in the path to be bypassed. Thus, if a client looks in the Stanford directory to resolve the name (STANFORD)ROOT.USA.MASSACHUSETTS.MIT.JULIET SUTHERLAND the MIT directory might contain an entry for the multiple component string ROOT.USA.MASSACHUSETTS.MIT which points directly to the MIT directory. This would allow the tree name to be resolved without searching either the root directory for the entry USA or the directory thus pointed to for the entry MASSACHUSETTS. A similar technique is used in the telephone network where a switch in Manhattan might resolve the telephone number 415-473-1234 by mapping the six digit sequence 415-473 into a direct trunk to the appropriate downtown San Francisco exchange. [19].

Let us summarize the discussion thus far. The components of a multi-component PersonnaName may have two very different interpretations. In one interpretation they are values for well-known and commonly understood attributes or fields in a name-address tuple. A directory database is a collection of tuples. In theory, no implication concerning the physical partitioning of the database is implied by the component structure. In practice, all directory databases are required to support a common minimal set of fields which are used to partition the name space among DSAs.

---

[8] *naming network:* a directory system in which a directory may contain the name of any object including another directory. An object is located by a multi-component path name relative to some arbitrary initial directory.

A second interpretation views a multi-component name as a series of entries in a network of directories. Each component is resolved within a specific directory. The resolution of the component is either a pointer to another directory or the name of the User Agent pointed to by the path name. A path name cannot be resolved without specifying the initial directory for resolution of the first component. Tree names are a special case of path names in which the naming network is a rooted tree. The directory for the first component is then well-known and can be found either because every directory has a pointer to the root or because every directory has a pointer to its parent.

The path name model for PersonnaNames, without restriction to a hierarchical naming network, provides great generality in the organization and distribution of directory service. Any service which maintains a table of PersonnaNames to User Agents or PersonnaNames to directory pointers can be part of a PersonnaName path name. Corporations, non-profit groups, municipalities, and service providers may all be providers of directory service. The set of entries stored in a directory need not be defined in any common manner—the M.I.T. directory may contain entries which are names of persons or entries which point to other directories. Path names are always resolvable by a directory service without user interaction.

If PersonnaNames are path names, then it is possible, even likely that many PersonnaNames will map into the same User Agent. Thus,

(ITU)USA.UNIVERSITIES.MIT.MARVIN SIRBU-HOME ADDRESS

is a synonym for

(ITU)USA.MASSACHUSETTS.CAMBRIDGE.MARVIN SIRBU

The former "name" can still be used to find Sirbu's home User Agent, even if he changes his residence from Cambridge to Boston, while the latter cannot unless historic information is kept in geographic directories. Far better might be the path name

(ITU)USA.UNIVERSITIES.MIT.ALUMNI.MARVIN SIRBU-HOME ADDRESS

which is likely to be usable even if Sirbu changes jobs in the future.

Users will thus learn to rely on path names which are unlikely to change. They may also choose to use path names for which the intervening directories charge the lowest price (Cf Section 5.1.4).

4.3. Directories versus Directory Service Agents

Each component in a path name is resolved in some directory. A directory is simply a table of bindings between names and objects. In order to assure availability, this table, or directory should be maintained on multiple service hosts, or by multiple Directory Service Agents. Thus a directory, and a DSA are not the same. A DSA may provide service for multiple directories. Conversely, any given directory should be replicated among several DSAs. The Xerox Clearinghouse architecture distinguishes Registries from Registration Servers in order to assure availability, and to share the processing load of handling directory requests among multiple servers. [13, 18]

4.4. Finding a PersonnaName

Finding an unambiguous PersonnaName for a correspondent differs from mapping PersonnaNames into User Agents. We expect the latter operation to proceed, in most cases, automatically. The former is a more difficult task, and is assumed to require an interactive dialog.

Whereas a path name approach is suitable for resolving a PersonnaName, finding a PersonnaName from general information about a user more strongly resembles identifying a name-address tuple using various attribute values for well-known fields. Interaction with the DSA should resemble a database query in which attributes of the desired record are provided until it is uniquely identified. As with any query system, it must provide a number of capabilities.

- The DSA must be able to provide the client with information from the data dictionary—*e.g.* the names of the attributes for which data values are available, and something about the nature of those attributes. Some of these attributes may be well known and commonly defined among DSAs. Common definition may imply a central source for the data dictionary, such as the CCITT, or it may imply agreements between organizations providing directory service. These are not mutually exclusive options.

- The DSA must be able to respond with the count of records which match a particular *attribute:value* query, and allow further searches on the subset thus obtained. Ideally, full boolean query support should be provided.

- If a limited thesaurus of values is defined for some attribute, this thesaurus must be obtainable.

- DSAs will have only part of the total name space of users. Most likely a DSA will have a partition corresponding to specified values of some attributes. The DSA should provide pointers to other DSAs which may have information on the desired set of records.

It is possible, of course, that clients will be able to guess a PersonnaName *a priori*. If you know that Marvin Sirbu graduated from MIT, you might guess that (MIT)ALUMNI.MARVIN SIRBU was a valid PersonnaName. However, this PersonnaName, is not resolvable unless you know where to access

the MIT directory. The name of a directory is a multi-component name of the form *e.g.* (ROOT)USA.UNIVERSITIES.MIT. One might try several names such as (ROOT)USA.ORGANIZATIONS.MIT or (USA)MASSACHUSETTS.MIT until one locates the MIT directory.

4.5. User Agent Names and Addresses

We have suggested above that directories map PersonnaNames into User Agent Names. Is such a binding necessary? If the "name" of my User Agent is always the same as my PersonnaName, no directory service is required for this level of binding. As long as we are certain that we will never want to have more than one User Agent associated with the same Personna, then using the same name will not pose any difficulties.[9] We have already noted that several forms of a PersonnaName may map into the same UA Name; thus, an equivalence of PersonnaNames and UA Names implies that a UA may have numerous aliases.

A UA Name which has the same form as a PersonnaName provides no hints as to where such a UA is located. A third binding *is* needed from the directory service, therefore, which maps a UA Name into a UA Address. There are in fact two forms of UA Address. From the point of view of the originating MTA, a UA Address is simply the name of the Message Transfer Agent which currently accepts mail for the recipient UA. From the point of view of the recipient MTA, the UA Address is a network attachment point to which mail for that UA should be directed. This corresponds to the difference between the "address" of a UA from the point of view of CCITT protocol P1 versus protocol P3 [3]. In particular, such a binding must indicate which Message Transfer Agent currently accepts mail for this User Agent.

An MTA in turn can be regarded as a service which runs on some host. As noted at the beginning of Section 4, at least 3 additional bindings are needed to map this service into a route over the internetwork. Viewing an MTA as a service, as opposed to as a host allows for the service to migrate to a different host in the event of hardware difficulties. Alternatively, one can imagine that MTAs are permanently bound to a particular host. In that case, reliable mail delivery might require that the mapping from UA to MTA specify secondary MTAs to use in the event the primary MTA is out of service; this approach is used in the Grapevine system [18]. When MTAs are viewed as a service which can be provided on different hosts, there is no advantage in having multiple bindings from a UA to several MTAs.

---

[9] The Xerox Ethernet standard uses the same 48 bit number for both host identifier and network attachment point identifier. This has led to problems when a host has more than one attachment point to the network (Cf. [17]).

5. Issues in Directory Service Design

In this section we attempt to provide a reasonably comprehensive list of directory design issues, both technical and organizational, which can be used as a starting point for further discussion. To this end, while we have much to say on some topics, others are included for completeness and are discussed only briefly.

5.1. Directory Operation

5.1.1. Direct vs Indirect

In obtaining a PersonnaName or an address, a client might interact with a directory service and the DSAs that implement it in either of two ways. One model puts the burden on the client. The client contacts a DSA; if its directory has all of the required information, the transaction is complete. If the directory has none, or only part, of the desired information, the client must contact another DSA. The name and address of the next DSA may be provided by the previous one or the client may determine them by other means. In any case, the client must establish direct contact with another DSA, and continue this pattern until a complete name or address is obtained. This approach is called "Direct Access" (Figure 5-1). [8]

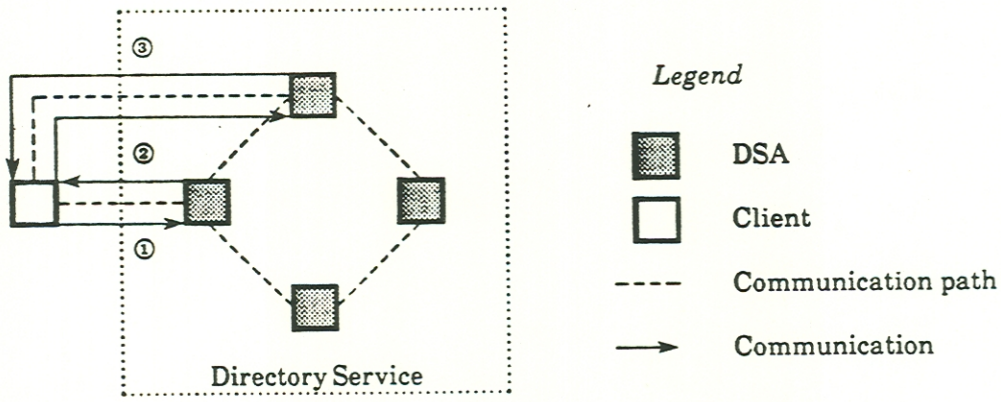

Figure 5-1: Direct Access to DSAs.

A second model has the first DSA making the contact with the next DSA, and so on, until the name is resolved. This model is known as "Indirect Access" (Figure 5-2). [8]

Domestic telephone directory assistance in the U.S. is provided by direct access. If a local directory assistance operator doesn't have the right part of the database (e.g. the city of the person being called) the operator will usually provide the correct number for directory assistance in that city. The user must then make a new call to the correct directory assistance service. International directory

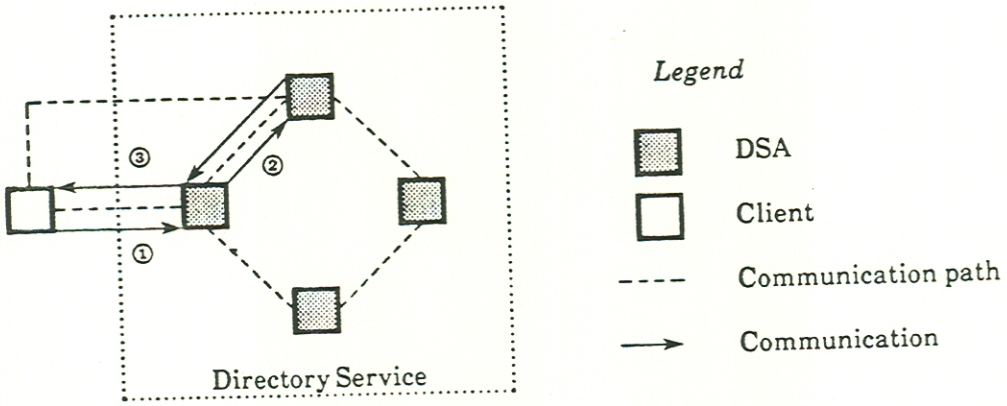

Figure 5-2: Indirect Access to DSAs.

assistance, however, is provided indirectly. The domestic operator establishes contact with the foreign directory assistance operator to resolve the query.

Direct access has the advantage that it allows the client to choose the communications network used to contact each DSA, thus permitting control over communication costs. In addition, if the client can communicate directly with every DSA, it reduces the processing load on DSAs acting as intermediaries.

Indirect access may offer economies of scale in communications if DSAs are in frequent contact. Billing may also be simplified through the use of a settlements process between DSAs. Direct connection requires direct billing to the client for DSA service.[10] Authentication may also be simpler with indirect access, since DSAs may have established procedures for authenticating themselves to each other and users would need to establish an initial authentication with only one DSA.

5.1.2. Incremental vs Absolute Name Resolution

There are two methods by which names can be resolved. One method specifies that the name-to-address mapping must be performed completely before a message is sent. That is, the message must be sent with a complete address. The other method allows the mapping to take place in incremental steps. The sending entity resolves enough components of the name to determine an address of the next forwarding point. The next entity resolves some of the remaining components in order to send the message further. By the time the message is delivered to its destination, the name has been completely resolved. We refer to the first method as *absolute* name resolution, and to the second method as *incremental* name resolution.

---

[10] Consider the overhead involved in maintaining account information for each possible user of a DSA.

Incremental resolution has the advantage that the name resolution and message transport can occur simultaneously. For absolute resolution, one might find the case where an expensive connection is required to perform the initial resolution and another expensive connection is required to send the message. CSNET[11] allows a form of incremental resolution to accommodate mail users who only connect to the network once a day. By using incremental resolution, these users avoid having to wait a day to receive the address before they can send the message. Incremental resolution may also be required if the sender cannot directly address the destination, for example due to differences in communications and addressing protocols.

One can argue, however, that caching—storing information locally—gives absolute resolution the advantage. With incremental resolution, one can only cache the first part of the name-to-address mapping, and later parts of that mapping must be performed each time a message is sent. Caching, combined with absolute resolution, allows the mapping to be performed only once with the cached address perhaps used many times before it is no longer valid.

A further problem with incremental resolution is that if the structure of the name and address are very different, incremental resolution may produce a very different route. In the worst case, incremental resolution of a name may send a message across the country and back, while the address might have shown that the physical destination of the message was nearby. In fact, incremental resolution may imply that the name and the route are related, while absolute resolution keeps the name and the route distinct.

If organizations are mutually suspicious, and if an address in some way reflects either the organization's network structure or organization chart, the organization may not want to divulge employees' complete addresses. The organization may choose instead to perform that part of the name to address mapping that occurs within its control. One can thus imagine a system whereby absolute resolution is used to contact the organization's gateway, which then performs any further mapping that may be required. Incremental resolution can also confine directory updating to within the organization.

These considerations, are more pertinant to the electronic mail system which uses the directory than to the directory itself. The mail system makes the choice between absolute and indirect name resolution, but this choice affects the directory service, as the discussion of the implications of caching shows. The issue of direct vs indirect DSA communication is only of concern if absolute

---

[11] Cf. Section 6.3.

name resolution is presumed.

5.1.3. Caching

Caching in directory systems has many permutations, depending on the answers to the following questions: *When* should caching occur? *Where* should caching occur? *What* should be cached? And *how* should the caching be performed? We will not answer the last question since a discussion of caching algorithms is beyond the scope of this paper.

By looking at the use of a directory service we can answer questions about when and what to cache. In obtaining an unknown name, a client will contact some directories and will make use of some attributes to describe the target of the query. Since the client will have obtained a unique name at the end of this process, the complete set of directory names and attributes should also uniquely identify the target. One might, therefore, consider caching the set of directory names and attributes. However, by the time that set is complete, one has also obtained a name. The only reason to save the information used to obtain the name would be to obtain the name again. Since names should change slowly, if at all, it seems unlikely that anyone would cache the information used to obtain the name. Having obtained a name, one might want to cache it to avoid having to get it again. Similarly, once an address has been obtained, one might want to cache it as well. We expect that different caching methods will be used for names and addresses since addresses will change more quickly than names.

Names and addresses can be cached at several locations. A directory might want to cache information from another directory. This would be particularly likely under the indirect model, since having information locally would save the cost of obtaining it from a remote directory.[12] Under the direct model, a directory might cache information, but it would be purely as a service to its clients. Names and addresses can be cached by a client or a client's UA. Caching at this point allows locally assigned nicknames and aliases. Finally, caching can be done by an MTA. An MTA might store the addresses for particularly common names, thereby saving the time and expense of using a directory to resolve them. Fortunately, all of these types of caching can coexist, since all are likely to be desirable, and each meets different needs.

There are two concepts within directories and mail systems that are strongly related to caching. The first concept is that of *authority*. Cached information, unlike information replicated for reliability purposes, is not guaranteed to be correct. There must be, however, a place where the information is always supposed to be correct. That is, there must be an authoritative directory to which one can

---

[12] This presumes that communication costs are more significant that storage costs.

refer. In addition to the concept of authority, there must be a way to determine that a cached item is, or may be, incorrect. This might be done via an error message, by a time out, or by some other method.

5.1.4. Accounting and Billing

There are three distinct models for how directory services will be financed: by the network, by the originator, and by the recipient. Examples of all three of these methods are in common use today.

1. Telephone Directory Assistance service has traditionally been furnished free by the communications network services provider. In theory, the cost of directory assistance is recovered from other telephone service charges.[13]

2. Recent studies by the phone company have shown that a small number of subscribers make large numbers of directory assistance calls. In order that these heavy users bear their appropriate share of Directory Assistance costs, phone companies have begun to charge subscribers explicitly for local calls to Directory Assistance above a specified maximum per month. Besides telephone callers, mailers frequently pay for address lists which resolve generalized directory queries such as "the addresses of all doctors in the Greater New York area".

3. The Yellow Pages is an example of a directory service paid for by the recipient of a communication. Also, the corporate telephone operator, who resolves requests for an individual's extension, is paid for by the receiving organization.

These examples suggest we should expect an equally diverse pattern of payments and billing in an electronic environment. Organizations which expect to benefit from making their members accessible will provide directory services for free. Network providers may also be expected to provide such services to their subscribers to induce additional network usage. Clients will be charged where the volume of queries is large, or a search is performed to retrieve a large set of addresses meeting specific criteria.

5.1.5. Update

Managing the addition, deletion, and change of directory entries, assuring the authenticity of update requests, and propagating changes to other directories storing the same information, are practical problems that must be addressed by any directory system. We will not discuss authentication methods, and have only one point to make regarding updates.

If an attribute or name component changes, it may have to change in several directories. Given

---

[13] AT&T has recently announced its intention to charge for the use of its network to gain direct access to remote Directory Assistance operators. [22]. The advent of competition means that AT&T cannot be assured of carrying the resultant call following a Directory Assistance inquiry.

multiple unaffiliated directory service providers, should it be the target's responsibility to see that the change is made in all relevant directories or should the update be propogated automatically by the directory service agents? Automatic propogation should lead to more reliable and consistent updating; however, such cooperation among directory service agents may be too much to expect in a competitve environment.

5.2. What Kind of Directory Information?

In its most basic form, a directory is simply a table matching names to addresses. As a minimum, then, a directory must contain names or components of names, and for each name component either an address or the name and/or address of a directory where that component can be resolved. In order to help a client find an unknown name, however, a directory may contain other kinds of information. Since this information is to be used to identify the name of the target, it will describe some aspect or *attribute* of the target. Different kinds of attributes will be used to identify different kinds of targets. For example, name, geographic location, and organization may be attributes associated with people, while name, topic, size, and membership may be attributes associated with distribution lists. As we have pointed out in Section 4.4, in order for attributes to be useful, a client must be able to determine what attributes a directory supports. This can be done by having all directories agree to support certain attributes, by having a protocol for obtaining attribute information, or by a combination of the two.

Issues of personal and organizational privacy become important in determining what types of attribute information to provide. Some of these issues are addressed in Section 5.3.

There are other kinds of information that a directory might provide as a service to its clients and because it seems a logical place within the message system to keep the information. Examples of this include tariff information, information about the capabilities of recipient's UAs, and authentication information. The scope of this type of information, and its relationship to names, addresses, and attributes are issues that have not been addressed by most directory efforts. Clearinghouse provides an arbitrary list of properties, values for which can be stored with each directory entry.

5.3. Types of Listing

In the US, most telephone directories distinguish between regular listings, unpublished listings, and unlisted telephone numbers. In a regular listing, a person's name, address, and telephone number appear in both the printed directory and the information operator's directory. For an unpublished listing, the information appears in the information operator's directory but not in the published directory. An unlisted telephone number does not appear in either the published directory or the

information operator's directory, although the fact that the person has a telephone does appear in the information operator's directory. Similar distinctions will apply to electronic mail directories.

Given the complex attribute data that may be maintained in a directory, there will be many more types of restricted listings. A person might specify that the values of certain attributes be listed as null. A person may specify a list of people who are allowed access to that person's address. An organization that provides optional attributes in its directory might want to distinguish between permanent employees, temporary employees (including consultants), friendly outsiders, and competitors, making a different amount or kind of information available to each. All of these examples can be provided by a simple structure.

For each attribute there can be several kinds of listing:

1. *Regular*	The value of this attribute is available to any client.
2. *Restricted*	The value of this attribute is available to a specified list or class of clients.
3. *Unlisted*	The value of this attribute is not available to any client and is always listed as null.
4. *Verified*	The value of this attribute can be verified but not volunteered. That is, if the client specifies the correct value for the attribute, the DSA will confirm that the value is correct. For example, the DSA might confirm that there is a Joe Smith at 56 Broad St.
5. *Volunteered*	The value of this attribute can be provided by the DSA as a choice to the client. For example, the DSA might ask if the client means the Joe Smith at 56 Broad St or the one at 21 Elm St.

Regular, restricted, and unlisted are mutually exclusive. The listing options for some attributes might be chosen by the person to whom the attribute refers or the options might be chosen by the organization maintaining the listing.

The difference between volunteered and verified information does not protect the privacy of the target from determined inquiry. A client who is willing to make a sufficient number of inquiries can obtain enough information in most cases to infer the values of attributes that can only be verified. This distinction, then, is useful to prevent everyone who makes a query from seeing certain information, but cannot protect against a client who is determined to obtain that information.

6. Existing Directory Systems

Having discussed a number of issues regarding the design of directory services and DSAs, we now examine some existing systems to see how they have resolved the questions we have raised. We will discuss three directory systems which are in various stages of design and implementation as well as the work done on models of directory service by IFIP Working Group 6.5.

Figure 6-1 summarizes our discussion of these systems and adds the telephone directory system for comparison. In the figure, the columns represent the answers to the following questions:

DSA Communication
: What method is used to obtain information from several directories? The possible values are direct, indirect, and Not Applicable.

Discovery
: Is the directory service designed to support a client in determining an unknown name? In the telephone directory system the yellow pages provide such support while the white pages do not.

DSA Architecture
: Which architecture is used to provide the directory service? The possible answers are distributed or centralized.

Caching
: Does the directory service allow caching? In those systems where caching is allowed, the method and place are described in the section that describes that system.

Access Control
: Does the directory service allow restriction of access to some information? Note that this is for read access and is not the same as authorization for update.

Authoritative
: How much of the information obtainable from a DSA is authoritative?

Name Form
: What type of name does the directory assume? The possibilities are tuples or some type of tree.

6.1. Clearinghouse

The Clearinghouse name server was designed at Xerox and is described in [13]. This directory maps names of the form "Individual@Domain@Organization" into sets of properties. Properties have names, types, and values. Properties can range from, for a printer, its network address, to, for a user, the name of the mail server that stores that user's mail. The examples and terminology of the names suggest that the designers intended "organization" to refer to a corporate entity and "domain" to the corporation's parts. However, one can generalize this three level structure. An "Organization" might be a state such as Massachusetts and a "Domain" might be a city such as Cambridge. Whatever the interpretation of the levels of names, the clearinghouse is explicitly designed to be a distributed system composed of many local clearinghouse servers.

	DSA Comm.	Discovery	DSA Architecture	Caching	Access Control	Authoritative	Name Form
Clearinghouse	direct	no	distrib.	unspec.	yes	all	3-level tree
CSNET	N.A.	yes	central	yes	no	all	tuple
ARPA	direct	no	distrib.	yes	no	some	multi-level tree
IFIP	either	yes	distrib.	unspec.	unspec.	unspec.	unspec.
Telephone	both	yellow pages	distrib.	yes	yes	all	tuple

Figure 6-1: Comparison of existing directory systems.

From the structure of the names, and the requirement that complete names must be used, it is clear that Clearinghouse is primarily intended to map names into addresses although it does provide some facilities for discovering unknown names.[14] These services, however, do not address the problems of personal privacy or corporate privacy and are only the roughest tools for finding addresses based on partially known information.

Clearinghouse servers are structured hierarchically, going from organizations to domains to individual targets. Organizations and domains have clearinghouse servers which are logically the principal servers for that level of the structure and which are required to know about lower levels of the structure. In addition, "each clearinghouse server points 'upward' to *every* organization clearinghouse." While the organization/domain hierarchy follows many of the same ideas that we have suggested, this final requirement quickly becomes difficult to implement on a nationwide, much less an international, scale. Consider the number of businesses that might maintain clearinghouse servers, combined with service providers for small business, geographical clearinghouses for residential use, and clearinghouses maintained by professional and social organizations. Consider also the communication cost, not to mention the administrative nightmare, of having every clearinghouse, regardless of its level in the hierarchy, keep track of every organization clearinghouse. Thus, while Clearinghouse will work within a single organization or a group of organizations, the requirement that all servers must know about all organization servers prevents Clearinghouse from being scalable to a national level.

Clearinghouse uses the direct method of searching for information. The "clearinghouse stub" maintained by each user contacts successive clearinghouse servers until it finds one that has the desired information. Given the strict hierarchy and the upward pointers, the clearinghouse stub contacts at most three servers. The final server performs any authentication and access control that may be required. Access control is performed at the domain level and at the property level. This allows managers of domain level servers to restrict access to all of their data and allows individual users to restrict access to information about certain properties.

---

[14] It is possible to ask about an individual via the *LookupIndividual* command and it is possible to list various types of information using the *LookupGeneric*, *EnumerateDomains* and *EnumerateOrganizations* commands. [12] The various lookup commands do simple string matching, while the enumerate commands simply provide complete lists of the domains and organizations known to the clearinghouse.

6.2. ARPA Name Server

The ARPA Internet is a network funded by the U.S. Department of Defense (DoD) which connects sites doing DoD research. These sites include universities, government contractors, and government agencies. A Request for Comment has recently been issued to the ARPA community regarding Domain Name Servers [10]. Through all of the following discussion, it should be remembered that these comments are based on a proposal which is still subject to change and which has not yet been implemented.

There will be many domain name servers, each knowing about a part of the total name space. The name space is defined to be a hierarchy of unlimited depth. The root of the tree is null and there is no discussion of how the first level will be defined. Initially, domain names and domain name servers will be used only for host names. Eventually information about user's mail addresses will be added. There is optional provision for the completion of partially known host names. The intention appears to be to provide a simple name/address mapping service.

A client will access domain name servers via the direct method. "If a name server is presented with a query for a domain name that is not within its authority, it may have the desired information, but it will also return a response that points toward an authoritative name server." In actual fact, a client will talk to a resolver, located on the client's host, which will then perform the necessary interaction with the name servers.

The only form of access control is the restriction that authoritative name servers may prohibit copying of any part of their databases by other servers.

The Domain name server proposal does discuss caching. Resolvers may cache address information. To maintain current information, data for caching is provided with a time-to-live after which it is discarded. Name servers may also cache information, in that they may store information for which they are not the authoritative source. In providing this information, a name server must also provide the address of the authoritative name server. The resolver must then treat this non-authoritative information as a hint which may or may not be correct.

6.3. CSNET Name Server

The CSNET is a computer communications network designed to connect computer science departments in colleges and universities throughout the United States. One of the services that CSNET provides is electronic mail, and in support of that service there is a name server [20]. The CSNET name server is centralized, existing at a well known address on the network. Issues, such as

direct vs indirect communication between DSAs, that apply only in distributed environments, are not relevant here.

The CSNET designers have not explicitly recognized that there are two levels of directory service. In discussing other work, they point out that a major difference between their work and the Clearinghouse project is the CSNET emphasis on facilitating lookup based on incomplete information. Since many of the CSNET name server goals involve user interaction, the design emphasis appears to be on providing a service to map attributes into names.

CSNET names, however, are not well defined. The CSNET name server makes provision for lookup based on a number of mandatory and optional attributes, known as "key words". Retrieval is based on string matching with partial strings permitted to allow for alternate spellings. Matching is done first on mandatory attributes, such as name and organization, with optional attributes used only to distinguish ambiguous results. The name server also provides a unique registration ID for each entry. The registration ID is used for "forwarding". That is, when a sender discovers that a recipient has moved, the sender's UA can use the registration ID to query the name server to obtain a new address.

The combination of mandatory and optional attributes that produces a unique result can be regarded as a name, as can the registration ID. The designers do not appear to distinguish between these types of names and, in fact, do not appear to regard the registration ID as a name at all. Since the registration ID is intended to be unvarying and can always be used to obtain an address, it most closely meets our definition of a name. The registration ID is neither easy to remember nor composed of multiple components, perhaps because it is intended only for machine use on a centralized server.

The CSNET name server does provide for caching of addresses by clients. How a client decides which addresses to keep is unspecified. When a message is returned as undeliverable, the client must query the directory to obtain the new address. This can be done automatically.

The CSNET name server provides for authentication for update but provides no access control for read-only operations. The designers assume that anyone providing information for the name server will provide only information that can be generally released. While this assumption may be valid in their environment of presumably friendly researchers, it does not generalize well to an environment of mutually suspicious organizations.

Unlike the other directory systems, the CSNET system allows the message to be sent with the query. If the query produces a unique result, the message is then sent to that address. If the query produces

an ambiguous result, the message is returned with the request for clarification. This option was provided for the convenience of hosts which have access to the network only once a day (via telephone) so that users would not have to wait for the result of the query to send the message.

6.4. IFIP

IFIP Working Group 6.5 has been in the forefront of conceptual work on message systems. We have adopted their model and terminology for message systems and have been inspired by their work on directories. They have suggested the concept of the DSA, have recognized that any directory system must be distributed, and have started to explore some of the issues that we have discussed in this paper. In particular, [8][15] has an excellent discussion of direct vs indirect DSA access. Working Group 6.5 has recognized the two level of mapping, distinguishing between a service to get a name and a service to get an address but the implications of this distinction have not been fully explored. Since Working Group 6.5 is just starting to address directory questions, many practical problems, such as management, access control, and accounting and billing have yet to be discussed.

7. Conclusion

The design and operation of a directory service for electronic messaging must meet a variety of criteria, some of which we have enumerated in this paper. At present no existing or proposed directory system meets all of the desired criteria. In particular, the directory problem must be understood as two separate problems: finding an unambiguous name for a user and finding the address which goes with that name. The structure and interpretation of user names is central to the design of a distributed directory service. The use of multi-component path names for users appears to allow many of the design objectives to be met. Further research is necessary on methods of updating directories, assuring availability through redundancy [18], and protection of personal and organizational privacy.

---

[15] The most recent subgroup meeting of WG 6.5 proposed numerous changes to this draft to be reflected in later versions.

References

1. Birrell, A.D., R. Levin, R.M. Needham, and M.D. Schroeder. "Grapevine: an exercise in distributed computing". *Comm. ACM 25*, 4 (April 1982), 260-274.

2. Casson, Lionel. "'It would be very nice if you sent me 200 drachmas'". *Smithsonian 14*, 1 (April 1983), 116-131.

3. CCITT Study Group VII/5. Message Handling Systems: System Model- Service Elements. Draft Recommendation X.400, International Telephone and Telegraph Consultative Committee (CCITT), Nov., 1983.

4. CCITT Study Group VII/5. Message Handling Systems: Message Transfer Layer. Draft Recommendation X.411, International Telephone and Telegraph Consultative Committee (CCITT), Nov., 1983.

5. Cunningham, I. *et al*. Emerging Protocols for Global Message Exchange. Compcom '82, IEEE Computer Society, Sept., 1982, pp. 153-161.

6. Deutsch, Debra. International Standardization of Message Transfer Protocols: An Overview. Compcom '82, IEEE Computer Society, Sept., 1982, pp. 162-167.

7. Haas, C. W. et al. "800 Service Using SPC Network Capability--Network Implementation and Administrative Functions". *BSTJ 61*, 7, Part 3 (September 1982), 1745-1757". Special issue "Stored Program Controlled Network".

8. IFIP Working Group 6.5. Naming, Addressing, and Directory Service for Message Handling Systems. Working Paper N78, IFIP Technical Committee 6, Working Group 6.5, Feb., 1983. Version 3.

9. ISO/TC97/SC16. Information Processing Systems - Open Systems Interconnection - Basic Reference Model. ISO 7498, ISO International Organization for Standardization Organization Internationale de Normalisation, 1983.

10. Mockapetris, P. Domain Names - Concepts and Facilities. RFC 882, University of Southern California, Information Sciences Institute, Nov., 1983.

11. Mockapetris, P. Domain Names - Implementation and Specification. RFC 883, University of Southern California, Information Sciences Institute, Nov., 1983.

12. Oppen, Derek C. and Yogen K. Dalal. The Clearinghouse: A Decentralized Agent for Locating Named Objects in a Distributed Environment. OPD-T8103, Xerox Corporation, Office Products Division, Oct., 1981.

13. Oppen, Derek C. and Yogen K. Dalal. "The Clearinghouse: A Decentralized Agent for Locating Named Objects in a Distributed Environment". *ACM Transactions on Office Information Systems 1*, 3 (July 1983).

14. Redell, David D. and James E. White. "Interconnecting Electronic Mail Systems". *Computer 16*, 9 (Sept. 1983), 55-63.

15. Ritchie, D.M., and K. Thompson. "The UNIX Time-Sharing System". *Comm. ACM 17*, 7 (July 1974), 365-375.

16. Saltzer, J. H. Naming and Binding of Objects. In *Operating Systems: An Advanced Course*, R. Bayer, Ed., Springer-Verlag, NY, 1978, ch. 3, pp. 99-208.

17. Saltzer, Jerome H. "On the Naming and Binding of Network Destinations". *Local Computer Networks* (1982).

18. Schroeder, Michael D., Andrew D. Birrell, and Roger M. Needham. "Experience with Grapevine: The Growth of a Distributed System". *ACM Transactions on Computer Systems 2*, 1 (Feb. 1984), 3-23.

19. Shoch, John F. Inter-Network Naming, Addressing, and Routing. Proceedings, COMPCON 78 Fall, IEEE, 1978, pp. 72-79.

20. Solomon, Marvin, Lawrence H. Landweber, and Donald Neuhengen. "The CSNET Name Server". *Computer Networks 6*, 3 (July 1982), 161-172.

21. Stoffels, Bob. "Turning Yellow into Gold". *Telephone Engineering and Management 87*, 19 (October 1 1983), 61-62. This article contains a reproduction of the 1878 New Haven District Telephone Co. Directory..

22. White, James A. and Jeanne Saddler. "A.T.&T. Long-Distance Unit Offers Plan for $1.75 Billion of Rate Cuts Next Year". *Wall Street Journal CCII*, 66 (Oct. 1983), 4.