LABORATORY FOR
COMPUTER SCIENCE

MASSACHUSETTS
INSTITUTE OF
TECHNOLOGY

MIT/LCS/TM-222

A TELEX GATEWAY FOR THE INTERNET

Friedrich Meier zu Sieker

May 1982

545 TECHNOLOGY SQUARE, CAMBRIDGE, MASSACHUSETTS 02139

MIT/LCS/TM-222

A TELEX GATEWAY FOR THE INTERNET

Friedrich Meier zu Sieker

May 1982

A TELEX GATEWAY FOR THE INTERNET

by

Friedrich Meier zu Sieker

May, 1982

This research was supported in part by the Defense Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under contract number N00014-75-C-0661.

Massachusetts Institute of Technology

Laboratory for Computer Science

Cambridge
Massachusetts

Abstract

Thesis Title: A Telex Gateway for the Internet
Author: Friedrich Meier zu Sieker
Thesis Adviser: Dr. Jerome Saltzer, Professor of Computer Science

The design of a gateway connecting one of the networks of the MIT Laboratory for Computer Science to the telex network is discussed. A description of the telex network is given. The relationship of the gateway to other resources of the network environment is considered to obtain directions for the implementation of new resources. The implementation of the gateway on the UNIX operating system outlined.

KEYWORDS: local computer networks
network interconnection
Telex

Contents

Section 1: Introduction

The Laboratory for Computer Science at MIT operates several computer networks. Some of these networks are connected by gateways. A gateway is a combination of hardware and software which allows the users of a particular network to exchange information with the users of a different network.

The topic of this thesis is the design of a gateway which links one of the local LCS networks to the public telex network. Telex is a worldwide network which supports the exchange of printed messages among approximately 500,000 terminals in about 200 countries. In many respects the telex network is similar to the telephone networks, the major difference being that telex circuits carry character coded messages whereas telephone circuits carry voice.

A telex gateway can provide quite useful services to the users of the LCS local networks. It enables a user to have messages forwarded to his lawyer in Boston, his broker in New York City, and his banker in Switzerland within a fraction of an hour. The gateway will also permit the user to send a telegram to his business partner who happens to be on vacation in Guatemala. While the user attends to other obligations the replies from his partners gather in his electronic

mailbox.

This thesis is a preliminary study with a twofold purpose. First, it is intended to be used as a guide to the implementors of the gateway. The tools available for the implementation are evaluated, the functions to be performed by the gateway are characterized, and structures performing these functions are suggested. Second, it attempts to produce considerations which are relevant not only to the application at hand but also to others.

The gateway is to be implemented on a PDP-11 minicomputer which is connected to both the Telex II and one of the local networks. The computer runs the UNIX operating system. The gateway will be a major subsystem of this environment. Its basic task is to accept messages from both networks and to arrange and control their proper delivery.

Although the gateway will perform the major functions involved in extending the telex service to the local users and, not to forget, accessibility of the local networks to telex users, it will only be part of a larger system which will be called the telex system. Part of the task of this thesis is to identify the tasks to be performed by the telex system and to assign them to the system's components, only one of which will be the gateway. Certain functions may require the participation of people.

In the following section descriptions of the local network, the UNIX host, and the telex environments are given. Telex is covered in some detail. In section 3 considerations regarding the user interface of the telex system are given. The stage is then set for a discussion of a division of labor between the gateway and the other system components in section 4. This discussion leads to an understanding of the functions the gateway should perform. These functions and their implementation are discussed in the remaining sections.

Section 2:

The Local Networks, the Gateway Host and Telex

- the local networks

There are six local networks and subnetworks; some are connected by gateways. The network classes are Ethernet, Ringnet, and Chaosnet. There is a gateway to the Arpanet. A large number of computers are attached to the networks. There are personal computers (Xerox Alto, IBM, Lisp Machines), minicomputers (pdp-11, Vax-11) and mainframes (pdp-10, HIS-6180). These machines run a variety of operating systems and subsystems. The heterogeneity of this environment reflects the diverse needs of the research groups which are using its resources either as tools or objects of study.

A user of any of these machines needs two crucial resources to be able to use the telex gateway. The first is a communication mechanism enabling the user to access the gateway. The second is software which enhances the bare communication mechanism such that comfortable interaction with the gateway is possible. This software will be called the 'command interpreter'.

Nearly all the machines in the network, including the gateway host, are able to exchange files using various file transfer

protocols. This file transfer capability is the lower level service upon which an electronic mail system has been implemented. The Arpanet mail protocol which sets a widely accepted standard specifies the format of mailfiles. This format is illustrated in appendix A. The protocol is specified in [1].

The protocol assumes the existence of programs which parse mailfiles to determine their addressees and invoke file transfer protocols to move the files among machines. Large computers usually run such a program as a continuous process called 'mail daemon'. These systems also feature programs which allow a user to generate, edit and manage mailfiles.

Offering both resources, the command interpreter and the communication capability, the mailsystem is a suitable vehicle for communication between user and gateway.

- the host environment

The gateway is to be implemented on a pdp-11 minicomputer which runs the UNIX operating system. UNIX will not be described at this point; its features are explained when needed in a later section. For a detailed introduction into UNIX the reader is referred to [2], [3], [4], [5].

The gateway host's primary communication mechanism is the trivial file transfer protocol (tftp) which is described in

[6]. It also runs a maildaemon. The central part of its mailsystem is the daemon's directory. This directory usually contains several request files which in turn contain the names of mailfiles ready to be delivered. Most request files are placed in the daemons directory by either a program called 'send' or by the machine's tftp daemon. 'Send' is invoked by UNIX host's users who wish to edit and send messages; tftp handles mail arriving from other machines in the network. However, any program may install request files in the daemon's directory.

After the daemon has processed all request files in his directory it 'goes to sleep', i.e. it inactivates itself. It is woken up again by the operating system after a certain period of time has elapsed or by a program like 'send' which signals the daemon after it has installed a request file.

- the telex network

--introduction

Telex is a worldwide circuit switched network which consists of a large number of subnetworks which are owned by independent organizations. In Europe these organizations are part of the governmental postal monopolies. In the United States telex consists of two subnetworks, Telex I, formerly called Telex, and Telex II, formerly called TWX. Both of these networks are owned by Western Union (WU). They differ

mainly in their transmission speed which is 50 baud for Telex I and 110 baud for Telex II. The telex networks in foreign countries follow the Telex I standard.

The American subnetworks are connected to the foreign subnetworks by gateways. There are five companies which share the right to carry international traffic. These are RCA, ITT, TRT, FTCC, and WUI (Western Union International). WUI is independent from WU; the latter is not permitted to carry international messages. WU has been granted the domestic monopoly by a ruling of the Federal Communications Commission. Despite of this fact gateway companies have begun to provide customers with dedicated lines which carry international traffic directly to their premises. There is the distinct possibility that future FCC rulings will remove restrictions from WU's and the international carriers' activities.

Telex is mostly used in the business world. A typical application is the forwarding of telegrams by the postal services. Press agencies use telex as a means to collect and distribute news items; travel agencies use it to obtain and to confirm reservations all around the globe. Especially in communication with foreign countries telex is the one public medium for fast and reliable message transfers.

--telex stations

A telex subnetwork can thought of as a giant switch which allocates communication channels between any two telex stations. These channels are usually referred to as 'calls'. Telex stations can be divided into four groups: regular stations, gateways, store and forward (s&f) servers and a group which offers information such as stock market reports and daily news.

A typical regular telex station consists of a teletype equipped with a telephone dialer and paper tape punch and reader. Every station can receive and print messages without operator assistance. The transmission of a message however requires an operator to control the station. This type of telex station is by far the most widely used. It is gradually replaced by micro processor controlled equipment with CRT's, electronic memory and a keyboard dialing capability.

A gateway permits an operator to obtain a call to a destination in a subnetwork other than his own. Gateways are equipped with computers which handle the interaction with the operator. The computers are backed up by gateway operators which switch themselves into a dialogue when problems arise.

A store and forward server accepts bunched messages and attempts to forward each of them for a certain number of times. It frees the user from the labor of coping with

congestion in the telex networks. Users of s&f servers receive short acknowledgements which indicate whether a server succeeded in its attempts to deliver a message or not.

International gateways provide s&f services; for example, they accept and deliver international telegrams. Some gateways also automatically offer s&f service during a dialogue with an operator if a desired call cannot immediately be allocated. Western Union's store and forward server is called Infomaster its services are summarized in appendix B.

--message transmission

A telex switch is somewhat similar to a telephone switch. In order to obtain a telex call a station performs an off-hook operation and transmits a telex number to the switch. After a time delay of about 1 to 5 seconds the originating station receives a call progress indication from the network. A list of these indications is given in appendix B.

Upon receiving a ring signal the answering station transmits its answerback. The answerback is a mnemonic sequence of about 10 characters which uniquely identifies a telex station. Telex stations exchange answerbacks for two reasons. First, the answerbacks are used to identify the distant station. Second, answerbacks are often exchanged before, during and after a message transfer to detect calls which were accidentally broken by a failure in the networks'

circuits.

A telex station transmits its answerback a) in reply to a ring from the network, b) upon receipt of the WRU (who are you) control character, c) if the opertor depresses the station's answerback key. An operator of a regular station may trigger the other party's answerback by transmitting the WRU character. WRUs are also contained in the answerbacks of some gateways. If they were permitted in regular stations' answerbacks, endless exchanges would result.

After the answering station has replied to the ring with its answerback the call is ready for operator-to-operator information transfer. Usually, the originating station transmits its own answerback before it starts with a message transmission or a dialogue with a gateway or s&f server. At any time may either station chose to release the call by performing an on-hook operation. Example 1 in appendix C shows a simple message transmission.

--interactive message transmission

The message of example 1 is simply recorded by the answering station. However, telex supports interactive traffic. Interaction between two operators is the exception rather than the rule because telex stations are often unattended. Nevertheless, an operator may try to initiate an interactive session by transmitting a control character which

rings a bell at the distant terminal where an attendant may decide to respond to it.

Usually interactive traffic occurs between a regular station and a gateway or a s&f server. The interaction patterns are very simple. Only when an operator at a gateway switches himself into a dialogue does the interaction become difficult to handle by software. Operators often employ a more or less standardized vocabulary of abbreviations. Examples 2 to 6 in appendix C illustrate various interactive sessions.

--directory assistance

WU provides computerized directory assistance. The procedures for obtaining it are extremely simple. Search keys are telex numbers, answerbacks or name and state of another telex user.

--billing

When a user orders a telex port Western Union sets up an account for the port under the user's name. There is no need to set up accounts with the gateway companies. They automatically generate accounts and invoices for their users.

The bills are itemized lists of telex calls. WU bills give the date, time and the number for each call. The times on WU bills are accurate only within minutes. Bills from gateway companies are more accurate. The date and time fields on their bills match those given during the calls. Date and time fields on gateway bills are part of a unique message ID which

gateways associate with every call. Most gateway companies also allow the user to include a departmental billing code in their messages which appears on the bill.

The telex carriers grant refunds for calls which were broken by a network failure. The customer simply mails a log of the interrupted call to the carrier who will credit the user's account. With Western Union this procedure does not work very smoothly, probably because WU does not supply unique message IDs which makes it hard for them to find a call in their records.

Section 3: The User Interface

To a large extent the sytem's user interface will be determined by the characteristics of the tools described in the previous section. Nevertheless it is necessary to consider the user interface from an a priori point of view in order to understand the restrictions which are imposed by the available tools.

--two user interface qualities: interaction and integration

There are two purposes for interaction between user and the telex system. First, the system should provide help and error messages which educate users about services, syntax, etc. Since an inexperienced user needs more aid than a proficient one, it is clear that the user should be able to gauge its extent. Second, it should provide status messages informing the user about the progress of his requests. The system should do its best to ensure that the user at some point knows precisely whether his message was in fact delivered or not.

The system's appearance should be integrated into that of other systems. Again there are two aspects demanding integration. Clearly, uniformity of syntax and procedures is a great convenience to the user. For example, sending mail

and telex should involve very similar commands. On the other hand, telex directory assistance should be obtainable with the same commands that are used to access other directory services.

Even more important, the user should be able to combine his capabilities in any reasonable way. Suppose for example he had access to a yellow-pages server, telex directory assistance, and to the gateway. It should then be painless task to find the telex address of companies identified by a yellow-pages search and to send enquiries to them via telex.

--types of users

There are two types of users: local users, telex users. In addition there will probably be an operator who monitors and assists the system. The interface the system should offer to the telex user is easily characterized: to telex users the system should look like an intelligent operator, not like a stupid machine; in other words, the system should be integrated into the telex user's view of the telex system.

This means nothing less than that the system must be able to handle all incoming messages, it may not reject any. It need not but perhaps should be able to handle interactive enquiries.

The operator's interface is not as clear yet. He should be able to monitor the traffic through the telex port. This will

be an invaluable debugging tool. No further attention will be given to this interface here because it strongly depends on the gateway design.

The local user's interface should of course possess the above two qualities. The user will also appreciate the presence of more concrete utilities. A very sensible one is a connection. A connection frees a user who frequently communicates with a certain party from going through the same repetitive procedure each time he wants to send another message. A user may send a telex by typing a connection name and a message body.

Apart from his routine interaction with the gateway the user must pay for the service he gets. Ideally he should be able to do this at his terminal. He should receive itemized bills; in addition summarized bills may be provided for project or department leaders.

- primary resources

Now that the user interface objectives and the tools available for the system's implementation have been outlined it is prudent to contemplate the process of matching the objectives with structures that are capable of meeting them.

The telex system will consist of a collection of resources each of which performs certain functions. Most of these resources were designed without consideration for their usefulness as components of the system. The telex system will appear as a client to these resources; they will be called 'primary resources'. In a figurative sense the system consists of slices of primary resources. Examples of primary resources are of course the mail system and the international gateways.

An ideal network environment consists of primary resources which can be configured to meet functional objectives without the need for new primary resources to be developed. All that it is needed is a manager who coordinates system components and takes care of functions arising from a specific configuration of resources. This manager is an example of a secondary resource.

We will now discuss some functions to be performed by the system and determine whether they can be performed by existing primary resources. This will lead to a clearer view of the gateway's management tasks. The discussion has the additional purpose to propose directions for new primary resources. The telex part of the system will not be considered because it must be accepted as it is.

- billing

The billing problem has two parts. On the one hand the billing procedures prescribed by the telex companies must be followed. On the other hand, users need to balance their accounts with the telex system.

The second part of the problem is sufficiently general to merit the implementation of a primary resource in the network environment. Both telex users and the system as well as other systems and their users would be the clients of this service. It essentially performs the functions of a banking and accounting service. It is clear that a consideration of the implementation goes far beyond the scope of this thesis.

The first part of the problem is in principle identical to the second, although it is somewhat less general since telex carriers are not prepared to interface to an electronic bank. Telex bills are not even available in computer readable form. Consequently assistance of people is necessary to

reconcile the system's and the telex carriers' accounts of the state of billing affairs.

We will consider neither part of the billing problem any further here. It is assumed that existing resources can be arranged to perform the billing functions. The system will simply generate a billing report for each chargeable transaction. The content of such a report is shown in appendix D.

- user interface: utilities and integration

The obvious resource where utilities like connections should be implemented is the command interpreter. The notion of a connection is not only applicable to the telex system but to other services as well. Its implementation essentially requires the command interpreter to have a macro processing capability.

Integration is very difficult to achieve. It requires a sophisticated interpreter which offers a uniform command language to the user and interfaces to network services via high level interpreter-server protocols. The interpreter translates the commands given by the user into sequences of the primitives specified in the protocol. It interprets responses received from a server to provide status messages for the user. The command interpreter enables the user to quickly generate, edit, transform and merge the datastructures

produced as outputs and required as inputs by the network servers.

The usefulness of such a command interpreter is plain. It moves the user interface implementation from the server into the user's machine. It would enable the user to access any network service which follows the high level protocol. In distributed environments where the user has access to remote services as well as to local computing resources such command interpreters are needed. We hardly need to comment on the difficulty of designing the high level protocols.

- Interaction

A desirable interaction pattern between a user and a network service consists of a stream of interaction units. Each unit consists of three incremental steps:

1. the user types a command
2. the system reponds with and indication
   of the command's result
3. the user examines the response and proceeds
   to step 1 or requests help

The user may at any time interrupt an interactive stream and turn his attention to another one. In a distributed environment this interaction pattern is extremely difficult to implement. The commands must be executed by different processor resources. This entails the following steps:

1. commands must be communicated to the resource
2. the resource must be allocated
3. the resource translates commands into a series of commands to other resources which implies steps 1 to 3

The primary resources available to the telex system cannot be configured to support the stream pattern. The file transfer mechanism is not suitable for fast communication of commands and responses between the command interpreter and the gateway. The resources on the telex side pose allocation problems.

First of all, the telex port may not be multiplexed among concurrent streams. The port must be allocated to one particular stream until the user decides to terminate it. This problem could be circumvented by employing several telex ports and allocating telex port streams to users. However, this would still not enable the user to suspend a certain stream.

Second, altough telex carriers do their best, they are often not able to allocate telex calls particularly to foreign countries. The system would then have to force the user to suspend a stream and, even worse, to continue it as soon as the call has come through. In addition, the allocation and management of gateway processor resources to multiple

concurrent instruction streams would pose substantial problems.

Consequently the characteristics of the available primary resources prohibit the implementation of an interactive stream user interface. Instead of single commands the user must send batches of commands and parameters to the gateway. A batch of commands will be called a request. A request must carry all the information needed to allocate the resources involved in servicing it.

The exchange of requests and replies can be supported by a new primary network resource which will be referred to as a template editor; It may work in the following way. A system like telex sends a template file to the template editor running on a certain user's machine. The file contains instructions, format specifications and text.

The template editor prompts the user for input according to the instructions in the file. It compares the input against the format specifications. If it detects an error it offers help text to the user. Serious consideration should be given to the implementation of such template editors. They would most likely find use in many applications.

A limited template facility which does not examine the users' inputs can be implemented with the existing resources. A telex user may be enabled to send a template request to the

telex system. In return he receives a mailfile which contains help text as well as labeled fields to be filled in by the user. A user may keep a set of frequently used templates in his command interpreter. An example of a template is given in appendix E.

Section 5: Gateway Structure

- general considerations

--types of gateways

So far it was assumed that the telex port is dedicated to the local user community. It is however entirely posssible that software systems will be developed which need to access a telex port in a way different from that of a user. For example, a network directory server may demand direct access to a telex port in order to provide optimal utility to his clients; delays in obtaining directory assistance seem to be less tolerable than delays in message forwarding.

It is possible to incorporate both a real time and a 'store and process' interface in the gateway. In essence, the gateway would have to be multiplexed among different types of users. A better solution is to implement independent gateways and to multiplex them among the telex ports.

A directory server is not available in the network environment. Since directory assistance is neccessary a pragmatic ought to be taken; the directory service should be integrated into the message forwarding software.

--objects

At a particular point in time the gateway will have a certain state which is recorded on the host's storage media. This state information can be partitioned into objects. Software design considerations suggest that an object should have a well defined life cycle: it is created, changes state and eventually terminates, i.e. its state information becomes useless garbage.

This means that the termination of an object should not depend on events whose occurrence is uncertain such as that of a user command. The gateway will then gradually move all objects out of its realm if no user commands arrive after a certain point in time. In a figurative sense, it will clean itself out. This ensures that no objects are lost inside the gateway, i.e. the users will always receive answers to their requests.

This displine provides another argument against implementing such notions as a connection in the gateway. A user will open a connection but he will probably forget to terminate it forcing the gateway to garbage collect upon such criteria as usage frequency.

The discipline can only be followed if the user issues batches of commands which are logically independent. Each batch can either be completely serviced or not at all. In a sense the

user's commands should be atomic. Thus software design considerations also seem to run contrary to the requirements of a interactive stream user interface.

- local requests

--service handlers

The mailfiles containing the batched commands from the telex system's users arrive in the gateway's mailbox. The gateway transforms these files into local request objects. There are several different types of requests:

- directory assistance requests
- template requests
- message requests

Message requests further subdivide into:

- domestic telegrams
- international telegrams
- mailgrams
- domestic Telex I
- domestic Telex II
- international telex

Each request type should be handled be a different software module; these modules will be called 'request handlers'. A request handler contains the procedural knowlege required to send a request via a particular telex service.

Message requests (with the exception of mailgrams and domestic telegrams) can be sent via different telex services. For example, international telexes can be sent via gateways, Infomaster and the gateway s&f services (Timetran in the case of ITT). Thus there may be several handlers for each type of message request.

It may prove convenient to further subdivide service handlers into two groups. One group handles traffic to unproblematic subnetworks (Europe), the other group handling traffic to exotic destinations like Guatemala.

Clearly, not all of these handlers must be implemented. All telex traffic can easily be handled by Infomaster alone. However, since Infomaster introduces time delay it is preferable to send messages via direct calls. Only if the direct call service handlers fail to send a message should it be submitted to Infomaster handlers. A minimal system should then include handlers for:

- WU Telex II to Telex I direct calls
- WU Telex II to Telex II direct calls
- RCA international direct calls
- RCA international telegrams
- WU directory assistance
- templates
- Infomaster services listed in

appendix B.

--request dispatching

Requests have to be dispatched to the appropriate service handler. This function can be performed by a software module which will be called the preprocessor. Apart from dispatching the preprocessor should also perform other functions.

It should transform the simple mailfile datastructure into a structure which is convenient to handle inside the gateway. For example help text ought to be removed from templates if requests are kept in core memory. The preprocessor may check whether a user is permitted to use the gateway, and whether he has supplied information such as account numbers and a reply address.

The preprocessor also converts the characters in the message body to the characters of the telex I and telex II alphabets. For example, telex I messages may not contain the character '{'. It must be converted to '('. If the preprocessor discovers syntactic or semantic errors it terminates the request. Otherwise it dispatches it.

The command batches should then be structured into two groups. The first group is parsed and interpreted by the preprocessor, the second by the service handlers. The first group should include the following fields:

From : [user name]

Reply_To : [user's network address]

Service : [service]

Country : [country]

Account : [user's account]

The first two fields are defined by the Arpanet mail protocol. The other fields are defined for purposes of this application. The preprocessor bases its dispatching decision on the content of the 'Service' and 'Country' fields. The [country] information enables the preprocessr to route requests to problematic countries to specialized service handlers or to the Infomaster handler.

--service handlers: conditions for terminating and suspending requests

Servicing and terminating template requests is straightforward. Templates requests can be serviced in a single pass through a service handler. A message request may require several passes through a service handler because a request must be suspended if one of the following conditions is met:

- a direct call can not be allocated, or a station is busy
- a broken call is detected
- a service handler is confused by

unexpected interaction

The first condition will often obtain if a call in a foreign subnetwork is sought. If a service handler is confused by the arrival of unexpected tokens a call can be suspended on the assumption that the deviation from the standard dialogue was caused by an operator. A supended request can be reprocessed by the suspending service handler after some time has elapsed. After a retry count exeeds a certain limit the request is submitted to one of the Infomaster handlers.

The service handlers should have two entry points. The first entry point accepts requests from the preprocessor; these requests are checked for syntactic and semantic correctness. The second entry point is for requests which have passed the checks as well as for suspended requests. Requests with syntactic or semantic errors are terminated.

During transmission of a message a service handler can recognize the following termination conditions:

- successful transmission
- receipt of status messages NA or NP
- successful transmission to a s&f service
  which does not provide acknowledgements
  via telex (mailgrams are not acknowledged
  via telex)

--waiting and matching

Requests containing messages which were accepted by a s&f service providing acknowledgments via telex cannot be terminated directly after transmission. These requests should leave the service handler and should be submitted to a waiting and matching module. This module terminates a request after it has matched it with an acknowledgement or after a certain period of time has elapsed.

--billing

The service handlers and the waiting and matching module generate billing requests for every chargeable interaction with telex.

--feedback

At various stages during the life cycle of a request it may prove sensible to send a status report to the user. For example, when a request is submitted to a s&f service, the user should perhaps receive a brief message. The precise pattern for these reports strongly depends on the dynamic behaviour of the working system. When a request is terminated the user should receive a log of the transmission activities which the service handlers performed on its behalf. If semantic or syntactic checks reveal an error the user should receive an explanation.

- telex requests

Requests arriving at the telex side of the gateway are more difficult to process than local requests. First of all the gateway has to handle telex requests in real time, they do not accumulate in a mailbox. Second, there is no standard format and syntax for telexes. Third, telex requests may not be rejected by the gateway.

--addressing

The gateway must parse telex requests in order to identify addressees. Because of the lack of a syntax this is a difficult task. The most accurate assertion regarding the format of a telex message is that it often can be divided into three parts: a header, the message body, and a trailer. These parts are usually seperated by two CRLF's. The header may contain keywords such as "ATTN" or "TO".

The addressing task could be simplified if it was possible to exert some influence on the format of telex messages. For example, at the very beginning of a transmission, i.e. after answerbacks are exchanged, the gateway could prompt the distant station with the following instructions:

THIS IS THE MIT COMPUTERIZED TELEX OPERATOR. PLEASE IDENTIFY THE ADDRESSEE OF YOUR MESSAGE IN THE FOLLOWING FORMAT:

ATTN:[TITLE] [FIRST NAME] [LAST NAME]

This strategy does not work if the distant station is computer controlled. Since a s&f service computer can be identified by way of its answerbacks the above lines could be suppressed in this case. However, there would still be a problem with computerized regular stations.

The real problem with this approach is its impracticality. An operator will be very reluctant to interrupt the message transfer and to reedit his message. He simply expects any message to be accepted. In short, this strategy violates the principle of integration.

Another possibility is to automatically transmit a reply address at the end of the transmission of every local request. For example the following lines might appear on a transmitted telex:

PLEASE USE THE FOLLOWING LINE IN ALL YOUR REPLIES:
ATTN: PROF. JOHN SCHWARTZ

If all communication via the telex gateway is restricted to be initiated by a local request there is a good chance that all arriving telex requests will contain a line marked "ATTN:". This restriction can be added by not including the gateway's telex address in any directory database.

It implies that no non-local telex user can use telex as a means for first contact with a local user. It therefore

limits the telex system's usefulness to individuals who enjoy a certain amount of publicity. Since these individuals may be reached by telephone or mail this limitation can be tolerated.

--addressing strategy

If this restriction is imposed the addressing algorithm can exploit the following circumstance: the answerback on a telex request has with all likelihood been received by the gateway during the transmission of a local request. It can also exploit the fact that it is probable that not more than one local user will communicate with a particular regular telex station. If this assumption does not hold, i.e. if more than one user communicates via a particular distant station, then the chance increases that the distant station will transmit properly addressed requests to the gateway.

If the gateway maintains a database which contains the names of all local users as well as the answerbacks received during transmission of their requests the gateway can use the answerback on a telex request as a key in a database search for the name of a local user. This search yields the name of a tentative addressee which may be compared to tokens found in a request's header.

If this method does not work the gateway can turn this strategy around and compare tentative addressee names found in the header against user entries in the database. The

tentative names can be identified by keywords such as "ATTN" and "TO". If no tentative name is found all the tokens in the header could be compared against all user names in the database.

The identification of answerbacks on telex requests is not a problem. The gateway may always trigger the distant answerback during direct calls. Store and forward servers do supply the answerback of the originating station at the beginning of a request.

Telexes which can not be addressed by the algorithm must be given to an operator. Telex requests which contain the bell control character signifying interaction with a distant operator should be submitted to the local operator without passing the addressing algorithm.

The addressing software should be structured into specialized service handlers and a general addressing module. The service handlers are capable of transforming the message formats used by gateways and s&f servers into uniform format which is processed by the general addressing module. For example, the Infomaster service handler extracts answerbacks from Infomaster messages; it recognizes and splits up bunched messages arriving from that server.

--the database

The database is updated by transmitting service handlers. It

can be used by the preprocessor to limit access to the gateway. In order to obtain access a user must have his name entered into the database.

To prevent it from steadily growing garbage collection is required. User names and inactive answerbacks need to be removed from the system. The simplest decision criterion for the removal of a name or an answerback is the time elapsed since it was last used. The existence of the database violates the object termination discipline. If telex numbers as well as answerbacks were kept, the database would contain much of the information needed to implement connections.

--on-the-fly addressing

While a telex request is being received the gateway could attempt to determine its addressee and inform the distant operator about the success of its efforts.

The usefulness of this service to the distant operator is questionable. Since successful delivery of telexes is more or less taken for granted the operator should only receive feedback in the rare event that a telex can with certainty not be forwarded. However, this certainty cannot be established by the addressing software; operator assistance is required (to make the event rare). Thus a useful indication of unsuccessful addressing efforts can not be given on-the-fly.

Section 5: Implementation on UNIX

A major task of the gateway is to muliplex requests among software resources and the telex ports (In this discussion two ports are assumed). In order to get an understanding of the multiplexing tasks the characteristics of UNIX processes and the telex ports must be examined.

- UNIX processes and the telex port

Since all requests are logically independent they can be serviced in parallel. In UNIX parallelism can be exploited by partitioning software into processes. UNIX attempts to exploit the parallelism inherent in the I/O activities of different processes. A process waiting for I/O to complete is kept in 'wait' state. After I/O is completed the process is moved to a 'ready' state. Processes in 'ready' state are serviced by the CPU such that interactive response times for future I/O activities are achieved.

There are two I/O calls: read(device, buffer, no. of bytes) and write(device, buffer). These calls are atomic in the sense that no I/O call accesses a memory location which is the source or the target of another pending I/O call. UNIX does not feature primitives which allow sequences of I/O calls to possess this atomicity. In other words, UNIX does not support locks.

The telex ports are accessed through read() and write() calls. A telex port must be dedicated to a transmitting or receiving process for the duration of several consecutive I/O calls. If two or more processes are permitted to use the same port some scheduling algorithm or locks need to be implemented.

- gateway processes

The question then is how to partition the gateway into processes such that parallelism is exploited, the burden on the host system is tolerable and the resulting design is simple. One approach consists of implementing one large process which services requests purely sequentially. This structure however exploits no parallelism at all.

A better scheme is to include a control structure inside the large process which multiplexes requests among software and telex port resources. This scheme has two serious drawbacks. Since UNIX processes are kept in 'wait' state until I/O is complete no I/O parallelism can be exploited with one process alone. It also means that a process which has issued a read(telex port) call may be kept waiting ad infinitum if expected input does not arrive.

Optimal parallelism would be achieved by dynamically allocating one process per request. Each process would remain active until it is able to terminate its request. This scheme is attractive not only because of the degree of parallelism it

achieves but also because only the telex ports need to be multiplexed by software which is not part of the operating system. Telex ports would have to be locked; a process ready to start telex I/O suspends itself for a period of time until a port becomes available.

An objection to this scheme is that it seems to claim too large a share of the host's resources. However, since the the telex ports are the bottlenecks most processes would be waiting for an unlocked port requiring very little CPU attention. Thus it is not clear whether dynamic process allocation would result in an unacceptable load on the host system.

A disadvantage of this scheme is the difficulty of monitoring it. During the implementation phase it will be necessary to stop, examine and restart the system. This is very difficult with a variable number of processes. For this reason a structure with a fixed number of processes ought to be chosen. The following configuration has a fixed number of processes which achieve sufficient parallelism:

Process A performing:

- preprocessing
- template handling

Process B performing:

- waiting and matching

Process C & D (one per port) performing:

- syntactic and semantic checks
- service handling
- billing request generation

Process F performing:

- addressing

These processes take requests form their input queues and processes them one at a time. Processes C and D should share their input queues. Both contain a manager who examines the queues to decide which request ought to be serviced after a previous request has been completed.

- recovery

In UNIX processes cannot share data segments in primary memory. This means that the input queues must be implemented as files. The queues can then also serve as checkpoints from which crashed processes are restarted. A file at a checkpoint should contain all state information which is needed for further processing. Actual processing is done on a copy of the request which is kept in core memory. When a request leaves a process a new file containing the new consistent state is created and the old file is discarded.

- reading the telex port

It was mentioned above that a read(telex port) I/O call may not return if expected input does not arrive. UNIX allocates

two invisible buffers for each telex port, one for input, the other for output. A system process fills the input buffer with characters arriving through a telex port.

When a user process issues a read(telex port) call the operating system suspends the process and starts a process which examines the buffer. If the buffer is empty the process suspends itself until it receives a signal from the process filling the buffer. It then moves the characters from the buffer to the user process's data segment; only after the characters have been transferred is the user process moved into 'ready' state.

Some UNIX systems feature a system call which returns the number of characters inside an input buffer enabling programs to do conditional read() calls. However the gateway host does not support this primitive. This means that another buffer must be implemented whose length can be determined by processes reading the telex port.

For two ports two buffers are needed. These buffers are updated by two new processes G and H. These processes repeatedly issue read(port, buffer, 1 byte) calls. Whenever a character is returned by the call it is appended to a buffer and a record containing the length of the buffer is incremented. Processes C or D examine this record, remove characters from the buffer and update the length record.

The buffer must be implemented as a file because processes cannot share data segments. A UNIX pipe cannot be used since a pipe can also not be read conditionally (a pipe inevitably synchronizes the processes reading and writing it). Implementing the buffer as a file does not pose an efficiency problem since all UNIX file I/O flows through primary memory caches.

- multiplexing the telex port

Processes G and H should also answer calls from the telex network. Incoming messages are recorded and placed into the input queue of the addressing module. Consequently both processes need to contain logic which recognizes whether characters arriving through the port are part of the transmission of a local request or whether they belong to an incoming telex request.

The following program illustrates the how the processes directly reading the telex ports switch between both modes. It is assumed that the host system is connected to a port modem which provides four indications regarding the state of the telex line:

^R - ring from a distant station
^H - distant station hung up
^A - distant station answers local call
^L - loss of call due to circuit failure

```
begin procedure read_port;

off_hook() ;    initialization sequence
on_hook() ;
unlock( port );
mode = 'idle' ;

read_loop: read( port, char, 1)

if char = '^L' then
   { if mode = 'distant' then
        { append_to_log_file( '^L' );
          unlock_port() ;
          close( log_file ) ;
          move_to_addressing_queue( log_file ) ;
        }
     if mode = 'local' then
        { append_to_buffer_file( '^L') ;
        }
     mode = 'idle' ;
     goto read_loop ;
    }

if char = '^R' then
   { if is_port_locked?() then
        { mode = 'idle' ;
          goto read_loop ;
        }
     else
        { mode = 'distant' ;
          lock_port() ;
          create( log_file ) ;
          off_hook() ;
          transmit_local_answerback() ;
          trigger_distant_answerback() ;
          goto read_loop ;
        }
   }

if char = '^H' then
   { if mode = 'distant' then
       { close( log_file ) ;
         move_to_addressing_queue( log_file ) ;
          unlock_port() ;
       }
     on_hook() ;
     mode = 'idle' ;
     goto read_loop
    }
```

```
if char = '^A' then
    { mode = 'local' ;
      goto read_loop ;
    }

if mode = 'distant' then
   { append_to_log_file( char );
     goto read_loop;
   }

if mode = 'local' then
   { lock( buffer_file ) ;
     append_to_buffer_file( char ) ;
     unlock( buffer_file ) ;
     goto read_loop ;
   }

end procedure read_port
```

This procedure never receives any control information from a service handler via read and write calls. It bases its decision where to route incoming characters upon the telex line status information provided by the modem. If '^A' is received the procedure knows that a service handler has seized the port and dialed up a remote telex station. The service handler controls the port by writing it directly and by reading incoming characters form the buffer file.

When a '^R' is received the procedure must first check whether a service handler has seized the port. This is necessary because a small period of time elapses between the time the '^R' is received and the time the operating system schedules the process containing the procedure to run. During this time span a running service handler may interrogate a telex port lock finding it idle although a distant station is ringing.

When the read_port procedure finds that the port was seized it assumes that the service handler has performed an off-hook, on-hook, off-hook sequence in order to preempt any ringing distant station. The procedure therefore switches to 'idle' mode. It will receive an '^A' when the distant station answers the service handler's call.

The above procedure assumes that all service handler calls are in fact answered. This is not the case. The procedure can be expanded to respond to a more elaborate set of status information from either the modem or the telex network.

The use of both the UNIX input buffer and a buffer file means there is an appreciable delay between the arrival of a character and the time it is recognized by a service handler. This time must be taken into account in the implementation of the service handlers.

Section 6: Conclusion

In a network environment the implementation of an application like the telex gateway needs to draw on primary resources which reside in remote host systems. The primary resources required by the telex gateway are the user command interpreter, a communication mechanism and a network billing resource. Also, authentication and encryption resources are needed.

There seem to be two models of interaction between distributed primary resources. On the one hand primary resources may interact by exchanging pieces of information in small increments; the interaction pattern is controlled by protocols. On the other hand resources may exchange chunks of information which were called requests. Requests carry all information needed to be processed by a remote resource.

As shown by the case study of the command interpreter to gateway interaction the first model poses substantial resource allocation and management problems. It therefore is necessary to adopt the second model in planning and implementing primary resources. This model implies the need for the management of request queues in front of and inside the network software resources.

It is therefore desirable that the operating system running the resource supports the queue management. It should enable the implementor to define request types and queues. The implementor then specifies template algorithms which the operating system applies to requests waiting queues. In other words, the operating system should remove the task of allocating processes to requests from the application.

UNIX is not an ideal operating system in this sense. Its approach is to provide the programmer with an extended machine instruction set to control its local resources; it confines itself to the management of the queues arising from the need to multiplex its own I/O devices.

Appendix A

Illustration of the Arpanet mail format

This is a simplified version of the example 3 given in section V.D in "Standard for the Format of Arpa Network Text Messages" by David H. Crocker et al., page 29, 21 November 1977. The document can be found in [1].

```
Date        : 27 Aug 1976
From        : Ken Davis <KDavis at Host_XY>
Subject     : status report
Sender      : Karen at Host_XY                    ;Ken's secretary
Reply-To    : Sam Irving at Host_XY
To          : George Jones at Host_AB
cc          : Sam Irving at Host_XY
Comment     : Sam is away on business.  He asked me to
              send this report.
In-Reply-To : < 1432.rty at Host_AB >
Message-ID  : < 4321.639.XYzi at Host_XY >
[carriage return]                                 ;null line
Since the last report dating 20 Aug ...           ;message text

[EOF]                                             ;end of file
```

Note: The contents of the "Sender" and the "Comments" fields are not intended to be interpreted by the software of a mail system. These two fields contain text from sender to receiver. Only the "Date" and "From" fields are required; all other fields are optional. The protocol permits user-defined fields; these fields must not bear names used by the standard.

Appendix B

- Infomaster Services

The following types of messages are forwarded:

- Telex I to Telex II
- Telex II to Telex I
- Telex I & II to foreign destinations
- domestic telegrams
- international telegrams
- mailgrams

Mailgrams are sent via telex to the US post office nearest to the destination; they are delivered by the postman. Telegrams are delivered by messenger. There is a single procedure for submitting the above message types to Infomaster.

- Call progress signals

--ASCII dialing option:

ABS - ring, no answer
DER - out of order
MOM - one moment please
NA - correspondence to this subscriber is not permitted
NC - no circuits
NP - invalid dial number; the party called is not or no longer a subscriber
RTO - interdigit timeout
OCC - busy

--touch tone and pulse dialing options:

As in the telephone network a call progress signal is provided. The intervals within which this signal is supplied indicates the status of the call:

Audible ring	10 signals per minute
Line busy	60 signals per minute
Reorder	120 signals per minute
Intercept	40 signals per minute

The customer may choose to receive the character sequences DER, NA, and NP instead of the intercept signal.

Appendix C

- Example 1: message transmission from a regular to another regular station; no interaction.

```
384873                          ;originating station dials
                                 Telex I number
JONESCO CAM                     ;receiver responds with
                                 answerback, call is open
WYMAN NYC                       ;originating station chooses
                                 to transmit its answerback
ATTN: KITTY THURBER             ;message begins
      SALES MANAGER

WE ARE DESPERATELY WAITING FOR A REPLY TO OUR
ORDER DATING 10 JAN 81

                    PETER WYMAN

JONESCO CAM                     ;originating station triggers
                                 distant answerback
WYMAN NYC                       ;originating station transmits
                                 its answerback
[on-hook]                       ;call is broken
```

- Example 2: simple interaction with gateway and message transmission.

```
5802294567                      ;Telex II number of gateway

RCA GA                          ;answerback and go ahead from
                                 gateway
85122222+                       ;country code and number of
                                 destination terminated with +
SMITH CO CGO                    ;gateway triggers originators
                                 answerback
JUL 09 1415 01234               ;gateway transmits message ID

JONESCO LDN                     ;receiver transmits answerback

TO: JOHN LINDGREN               ;originator transmits message
SUBJECT: PR MEETING

JOHN,
AN URGENT MEETING HAS BEEN SCHEDULED ...

  .....                         ;originator sends five periods
                                 to indicate end of message  to
                                 the gateway
0000.7                          ;gateway gives chargeable time

GA                               and go ahead for next message

[on-hook]                       ;originating station terminates
                                 call to gateway
```

- Example 3: interaction with gateway operator.

103	;station dials up ITT gateway
ITT GA	;go ahead from gateway
(15042)7161191+	;station transmits departemental billing code followed by telex number of distant station
MIT CAM	;ITT identifies originating station
11 18 1223	;gateway gives date and time
MOM FOR OPERATOR	;gateway asks originator to wait a moment for operator assistance
ITT TLX OPR F UI	;gateway operator
TRYING TO REACH IN CHINA 7161191	;originator
WHATS THE FULL ANSWBACK	;gateway operator
1191 SHANGHAI	;originator
NBR XXX THATS THE FULL ANWEBACK ???	;gateway operator
I WILL CHECK THROUGH INFORMATION	;originator
FINE TU BIBI	;gateway operator terminates call
[on-hook]	

- Example 4: gateway offers s&f services because
circuits are busy;the acknowledgement
is given in example 7.

```
103                              ;originator dials gateway

ITT GA                           ;go ahead

(87704)85616192+                 ;departmental billing code and
                                  telex number of destination

MIT CAM                          ;gateway triggers answerback

 11 04 1428                      ;gateway gives date and time

NCH                              ;subscriber's number has been
                                  changed
MOM                              ;wait a moment

TIMETRAN                         ;gatway's s&f server signs on
 PLS TYPE:OWN TELEX NBR
NEXT LINE: TEXT

921473                           ;originator types his number to
                                  enable TIMETRAN to send an
                                  acknowledgement
TO: TORE KNUTSEN
    NORSK DATA
    OSLO, NORWAY

RECEIVED YOUR LETTER ....        ;text


REGARDS,
A. E. SOERENSON
PROF. OF ELECTRICAL ENGINEERING
MIT
TELEX 921473 MIT CAM

CALL ACCEPTED                    ;TIMETRAN accepts message and gives
 JCU0918                          a message ID
[on-hook]
```

- Example 5: message transmission via Infomaster;
the delivered message is given in
example 8; the message goes from a
Telex II (TWX) to a Telex I (Telex) station.

```
9104201212                      ;Infomaster's Telex II number

WU INFOMASTER                   ;Infomaster's answerback

ABC CORP                        ;Infomaster triggers answerback

007436I020 1250EST              ;message ID and time

[control_A]                     ;originator marks address fields
01 CAMBRIDGE, MASS, JANURARY 20, 1982  ;first message,
                                 originator's city and state, date
TLX 921473 MIT CAM              ;address of destination station
ATTN JIM SNYDER                 ;addressee

[control_B]                     ;originator marks begin of text

DEAR JIM
REFERRING TO YOUR TELEX...
[control_C]                     ;originator marks end of text
ACCEPTED                        ;Infomaster accepts message
00001                           ;first message
[on-hook]                       ;originator terminates call
```

- Example 6: interactive enquiry from a distant station.

```
MIT CAM

GNET REG EGW                    ;distant answerback

IS THIS MITINP IN CAMBRIDE MASS

[bell] [bell] [bell] [bell]

NO THIS IS MASSACHUSETTS INSTITUTE OF TECHNOLOGY

THANK YOU
[on-hook]
```

- Example 7: acknowledgement for message of example 4.

```
MITCM004                        ;answerback of station sending
                                 the acknowledgement
 NOV 05 1981 0603               ;date and time

MIT CAM                         ;receiver's answerback


TIMETRAN
YOUR 11 04 1428 REF JCU0918 TO:
85616192
UNABLE TO DELIVER
NO RESPONSE - DEPT. (87704)

[on-hook]
```

- Example 8: delivery of message of example 5.

```
MIT CAM

WU INFOMASTER 1-007436I020 01/20/82
TWX ABC CORP                    ;Infomaster gives message ID, date,
                                 subnetwork and answerback of
                                 originator

ZCZC01 CAMBRIDGE MASS, JANURARY 20, 1982
TLX 921473 MIT CAM              ;ZCZC marks address block for
ATTN JIM SNYDER                  Telex I Infomaster messages

BT                              ;BT marks begin of text

DEAR JIM
REFERRING TO YOUR TELEX...

NNNN                            ;NNNN marks end of message
1254 EST

MIT CAM

[on-hook]
```

Appendix D

- data items in a billing request generated by the gateway

user name

user's account

user's network address

departmental billing code

date and time of request transmission

date and time supplied by carrier

names of carriers involved

message ID supplied by carrier

chargeable time given by carrier

cost estimated by gateway

log of the transmission if refund is to be obtained

Appendix E

Template for international traffic:

Date : 01-12-82

From : telex at CSR

To : John Doe at XX

[CR]

This is the template file you requested. Fill in all the fields under the dotted line according to the instructions. Then mail the everything under the dotted line to 'telex at CSR'

.....

service : Telex I

In the next field fill in the name if the country of the destination.

country :

In the next field fill in the destination's telex number; do not include the country code !

number :

In the next field fill in the destination's answerback.

answerback :

In the next field fill in your account number.

account :

Now, on a new line start typing your message!.....

References

[1] Arpanet Protocol Handbook, Network Information Center, SRI International, Menlo Park CA., Rev Jan 1978.

[2] D.M. Ritchie and K. Thompson, "The UNIX Time-Sharing System," Bell System Technical Journal, Vol. 57, pp. 1905-1929, July 1978.

[3] S.R. Bourne, "UNIX Time-Sharing System:The UNIX Shell," B.S.T.J, Vol. 57, pp. 1971-1990, July 1978.

[4] K. Thompson, "UNIX Time-Sharing System: UNIX Implementation," B.S.T.J, Vol. 57, pp. 1931-1946, July 1978.

[5] D.M. Ritchie, "UNIX Time-Sharing System:A Retrospective," B.S.T.J., Vol. 57, pp. 1947-1969, July 1978 .

[6] K.R. Sollins, "The TFTP Protocol (Revision 2)," in 'The Internet Protocol Transition Handbook,' Network Information Center, SRI International, Menlo Park CA., March 1982.