NASA-CR-194877

NASA
National Aeronautics and Space Administration

Report Documentation Page

IN-82-CR

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
		203552 13P

4. Title and Subtitle	5. Report Date
AstroMail: Electronic Mail for the Astrophysics Community	May 20, 1993
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Phillip H Scherrer, Richard S. Bogart	506-59-11
	10. Work Unit No.
	NAS5-30428

9. Performing Organization Name and Address	11. Contract or Grant No.
Center for Space Science and Astrophysics Department of Electrical Engineering Stanford University Stanford, CA 94305-4055	Final
	13. Type of Report and Period Covered

12. Sponsoring Agency Name and Address	14. Sponsoring Agency Code
Center of Excellence in Space Data and Information Sciences (CESDIS) Code 930.5 Goddard Space Flight Center Greenbelt, MD 20771	

15. Supplementary Notes

CESDIS is operated by Universities Space Research Association (USRA), American City Building, Suite 212, 10227 Wincopin Circle, Columbia, MD 21044

16. Abstract

As part of the NASA Science Internet User Support Services program, NASA Goddard was interested in R&D which could extend the SolarMail system developed by members of the Wilcox Space Observatory at Stanford University to support a larger astrophysics user community. Specific objectives of the R&D effort were to include:

- A clone of the existing SolarMail system with additional documentation, enabling a parallel mail system to be established by populating the database.
- A cloned version of SolarMail functioning with a user database similar to that of the High Energy Astrophysics Division (HEAD) of the American Astronomical Society.
- A report on the status and surveyed usage of SolarMail and its clones into an extendable distributed mail system to serve as the basis for AstroMail, including a draft declaration of policy.
- A prototype AstroMail system based on the above specifications and including at least SolarMail and one of its clones supporting a set of astronomy user databases as subsets.
- A report on the status of the prototype AstroMail with recommendations for future modifications to AstroMail.

17. Key Words (suggested by Author(s))	18. Distribution Statement
electronic mail networks	unclassified - unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
		13	

NASA FORM 1626 OCT 86

AstroMail objectives

AstroMail was proposed as a suite of user services designed to facilitate and enhance the value of electronic communication among members of the Astronomy, AstroPhysics, and Space Science research communities. This system was to be based on SolarMail - an existing mail service widely used by the Solar Physics research community for the last six years. The proposal described various enhancements to SolarMail and outlined the issues that needed to be tackled before building a prototype for AstroMail.

The various ideas and modifications described in the proposal centered around two core issues. The first issue was one of storing information in an organized manner so that the system was more reliable and robust as compared to its predecessor. The second issue was that of managing the stored information so as to ease the burden on individuals administering the system. By providing a structural framework which dealt with these core issues in an effective manner; it was reasoned that the provision of user services would be easier.

The following services were considered essential to the AstroMail system.

- Simple syntax email forwarding

  The objective was to provide for a uniform syntax for mail addresses that users would find convenient to apply in a variety of settings. Such an address form would help in the transparent handling of foreign systems and provide support for forwarding and replying. Though not quite intended, it was hoped that this would provide the basis for a natural evolution to ISO standards in the future.

- Mail exploders

  The objective was to provide for the dissemination of information to a pre-specified list of users. This could therefore be used to help in the distribution of newsletters, bulletins and alerts. Support was to be provided for multiple lists, on both a short-term and a long-term basis. The control over the membership was to be delegated to group leaders and project managers.

- Directory Information Management

  In addition to the message handling component, AstroMail was to provide for a Directory component. This Directory was to be a repository of information about users, projects, and correspondence. Individual users were to have the ability to manage personal information on their own. Further, they were to have the ability to screen the information that they would like to be made available to others.

- Bulletin Board Services

  In addition to offering active correspondence to the users, AstroMail was supposed to offer passive correspondence in the form of bulletin boards. Besides providing for a record of activity, it was hoped that it would provide a forum for discussion of ideas and scientific thought.

Tackling the core issues

Before going on to the issue of storing the information, it was necessary to identify the various kinds of information that needed to be stored. We identified the following categories of information that needed storage.

- User information

  A list of attributes were assigned to each user in the system. The attribute values were to be provided by the user, who also had control over how the information would be used. This list of attributes is listed in the appendices.

- Group information

  Each group had an administrator, who had control over the membership and the information maintained by the group. A group could have sub-groups, each with its own administrator. A list of group attributes are listed in the appendices.

- Logging information

  For purposes of administration, separate from user and group logs, it was found necessary to record exceptional events. A list of relevant attributes are listed in the appendices.

Given the above information that needed storage, the options regarding storage were examined. Each of these options led to differing structures for the system; each of which were evaluated.

There were three approaches that were considered :

1) Indexed Storage scheme

   This scheme was developed by Rick Bogart, and was based on storing information in the form of records that could be identified uniquely by an index. Such a scheme required developing software and routines which would manage the information on their own.

2) Database Storage schemes

   This scheme was initially based on storing the information in a commercial database, and using queries to retrieve information as necessary. Such a framework would enable rapid access to complex information.

3) Directory Tree Storage

   This scheme organized all information on a tree structure. The primary advantage of this scheme would be the ability to distribute the information over multiple machines. Consequently, delegation of responsibility for maintenance would be a feasibility. Further, evolution to ISO standards would be considerably easier in the future.

• Modifications to the Information Management scheme

There were three different approaches that were considered.

1) Email based schemes

This alternative was developed around the idea of using electronic mail for all information management. An automatic email processor was partially developed to aid in deciphering user requests and information that the user would provide. This option would have led to inflexibility in the use of email formats. The only potential advantage to such a scheme would be the benefit of the medium of electronic mail as the least common denominator.

2) Guest shell scheme

A guest shell was developed by Brian Frasca to enable users to modify and access information uniformly. This shell could be used by any registered user who could avail of "telnet"or "rlogin" facilities. The shell was entirely menu-driven, and could provide extensive help facilities to new users and had browsing capabilities. This shell could provide the basis for more sophisticated interfaces that could be developed later on.

3) Directory user scheme

Given previous Directory based schemes, software was available for modification so as to customize handling of the information. Besides providing X-based interfaces and window displays, this scheme also supported simple line interfaces. The primary advantage of this scheme was the manner in which information could be accessed in a transparent manner from various machines on the network. Further, elaborate schemes were available that provided for access control lists, and different views of the same data.

Building the proposed services

During the course of the project, multiple approaches to achieving objectives were implemented. The aims of mail forwarding and mail exploding both depend upon the manner in which the address are managed. Hence, it is essential that a solid framework be established for a robust and efficient address management scheme.

Address management was previously done by maintaining an ASCII file containing the SolarMail name, and the real address to which mail should be forwarded. This file was read by the mail transport agent, namely "sendmail" and then converted into a random access database of aliases. Thus, mail explosion and mail forwarding were treated uniformly as a method of recursively transforming a single name into a list of real addresses. Sendmail would then do the appropriate mailing to the addresses in the so formed list.

• Modifications to the address management scheme

There were three different approaches that were implemented.

1) Local Mail Handling.

A local mailer, "lxmail" was developed by Scott Rogers. This software, as we understand it, provided for a special address syntax which facilitated identification and therefore, special mail handling by "lxmail". On identification, the mailer would examine a particular database created from a flat file -maintained by the group manager/postmaster- and then perform appropriate mapping. After the mapping stage, the mail along with the recipient addresses and other relevant information would be handed over to the mail transport agent.

2) Central Mail Handling.

This approach used a slightly different mail transport agent i.e., Sendmail, reinforced with the IDA kit. The idea was to have all mail directed to users or aliases in a certain domain, say "science.nasa.gov" to be handled by a central machine. This machine would run the version of the mail transport agent, as developed at Stanford and would examine all envelope addresses. Based on the envelope address and a local database of address mappings, this central mailing agent would forward the mail to other machines which would then handle the actual mailing. It appears that such a scheme is supportive of the "distribution" paradigm underlying the initial proposal.

3) Directory Based Handling.

This approach relied on utilizing an X.500 based mail transport agent. The basic difference between this method and the previous methods is the manner in which address mapping is done. In the above methods, such mapping is done by a lookup into a table that is maintained at the local machine by the mail transport agent. In this scheme, the mailing agent instead of doing the lookup on its own, performs a query to a Directory Server Agent (DSA). This DSA then acts upon the query to determine what the recipient address ought to be. It is not necessary that the Directory be located on the same machine. Further, it may even be spread over a number of machines. On receiving the actual address from the DSA, the mail transport agent then rewrites the initial address, and performs mail transfer.

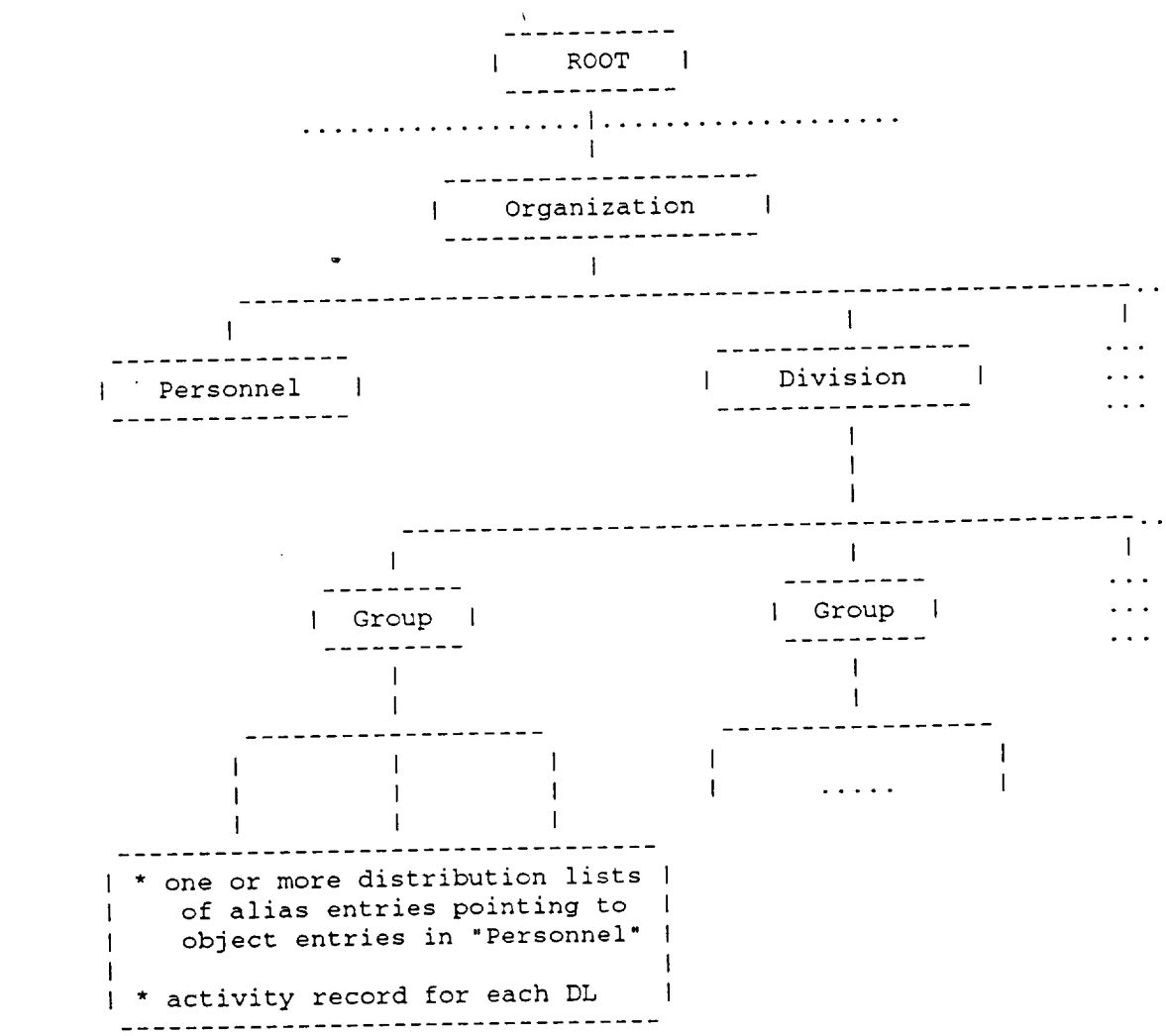
Directory Information Management was intended to reduce the load on the system manager, so that users could maintain their personal information in a convenient manner -on their own. It was upto the group manager/postmaster to decide whom to include in the mailings to the group. The basis of any directory is its content, its storage and the mechanism of its access. It was therefore, essential to re-examine and perhaps improve on the SolarMail scheme.

Future Development

The aims of the initial proposal are best achieved through utilizing a Directory structure. Besides providing for distribution of data, the elaborate security mechanisms offer a robust and reliable framework.

Appendix 1

Directory Information Tree :



Appendix 2

Information in the DIT :

We mainly use the following base object classes.

* top
  This is the "root" object class. All other object classes except the "alias" object class form subclasses of "top".

* alias
  This is a special object class to indicate that the entry is a pointer to some real object.

The following are the object classes being used.

* person
  SUBCLASS OF top
  MUST CONTAIN {
    commonName,
    surname}
  MAY CONTAIN {
    personAttributeSet,
    telecommunicationAttributeSet,
    postalAttributeSet}

* organization
  SUBCLASS OF top
  MUST CONTAIN {
    organizationName}
  MAY CONTAIN {
    organizationalAttributeSet}

* organizationalUnit
  SUBCLASS OF top
  MUST CONTAIN {
    organizationalUnitName}
  MAY CONTAIN {
    organizationalAttributeSet}

* mhsDistributionList
  SUBCLASS OF top
  MUST CONTAIN {
    commonName,
    mhsDlSubmitPermissions,
    mhsOrAddresses}
  MAY CONTAIN {
    description,
    organization,
    organizationalUnitName,
    owner,
    mhsDeliverableContentTypes,
    mhsDlMembers,

```
        mhsPreferredDeliveryMethods)

* applicationEntity
    SUBCLASS OF top
    MUST CONTAIN {
         commonName,
         presentationAddress}
    MAY CONTAIN {
         description,
         localeAttributeSet,
         organizationName,
         organizationalUnitName,
         supportedApplicationContext}

* dsa
    SUBCLASS OF applicationEntity
    MAY CONTAIN {
         knowledgeInformation}

* mhsMessageStore
    SUBCLASS OF applicationEntity
    MAY CONTAIN {
         description,
         owner,
         mhsSupportedOptionalAttributes,
         mhsSupportedAutomaticActions,
         mhsSupportedContentTypes}

* mhsMessageTransferAgent
    SUBCLASS OF applicationEntity
    MAY CONTAIN {
         description,
         owner}
```

We thus have the following heirarchy of object classes.

```
-alias
-top
   -person
   -organization
   -organizationalUnitName
   -mhsDistributionList
   -applicationEntity
      -dsa
      -mhsMessageStore
      -mhsMessageTransportAgent
```

Appendix 3

Object Class Definitions

```
# format:-
#
# name:                          oid:                    hiearchy: must :top may
#
# Set definitions:

telecommunicationAttributeSet = facsimileNumber, \
    telephoneNumber, telexNumber, \
    preferredDeliveryMethod

postalAttributeSet = physicalDeliveryOfficeName, postalAddress, \
    postalCode, postOfficeBox, streetAddress

localeAttributeSet = localityName, stateOrProvinceName, streetAddress

organizationalAttributeSet = description, localeAttributeSet, \
    postalAttributeSet, telecommunicationAttributeSet, \
    searchGuide, userPassword

personAttributeSet = alternateName, userid, emailAddress, \
    alternateEmailAddress, shellAccessLanguage, subscriptions, \
    memberships

groupAttributeSet = description, members, activityRecord, \
    subGroups

# Object Classes

top:                           standardObjectClass.0 : : objectClass :

alias:                         standardObjectClass.1 : top : aliasedObjectName :

organization:                  standardObjectClass.4 : top : O : \
    organizationalAttributeSet

organizationalUnit: standardObjectClass.5 : top : : \
    OU, FolderName, organizationalAttributeSet, \
    groupAttributeSet

person:                        standardObjectClass.6 : top : CN ,surname : \
    description, telephoneNumber, userPassword

organizationalPerson:          standardObjectClass.7 : person : : \
    localeAttributeSet, OU, personAttributeSet, \
    postalAttributeSet, telecommunicationAttributeSet ,title
```

```
applicationEntity:  standardObjectClass.12 : top : CN, \
    presentationAddress: \
    description, localityName, O, \
    OU, supportedApplicationContext

dSA:                            standardObjectClass.13 : applicationEntity : : \
    knowledgeInformation

# QUIPU defined object classes

quipuDSA:               quipuObjectClass.1 : dSA : \
    acl, edbinfo, userPassword, manager, quipuVersion : \
    description, lastModifiedBy, lastModifiedTime, \
    dsaDefaultSecurityPolicy, dsaPermittedSecurityPolicy, relayDSA, \
    listenAddress

quipuObject:                    quipuObjectClass.2 : top : acl : \
    lastModifiedBy, lastModifiedTime, entrySecurityPolicy

quipuNonLeafObject:     quipuObjectClass.6 : quipuObject : masterDSA : \
    slaveDSA, treeStructure, inheritedAttribute

quipuSecurityUser:  quipuObjectClass.7 : quipuObject : protectedPassword :

iSODEApplicationEntity: quipuObjectClass.8 : applicationEntity : execVector :

externalNonLeafObject:  quipuObjectClass.9 : quipuObject :: \
    subordinateReference, crossReference, nssr

# end-of-object-classes
```

Appendix 4

User and Group Attribute Definitions

```
# format:-
#
# name:                                oid              :syntax
#

objectClass:                           attributeType.0          :ObjectClass
aliasedObjectName:                attributeType.1          :DN
knowledgeInformation:                  attributeType.2          :CaseIgnoreString
Name,cn:                          attributeType.3          :CaseIgnoreString
surname,sn:                            attributeType.4          :CaseIgnoreString
localityName,l:                        attributeType.7          :CaseIgnoreString
stateOrProvinceName,st:                attributeType.8          :CaseIgnoreString
streetAddress:                         attributeType.9          :CaseIgnoreString
Discipline,o:                          attributeType.10         :CaseIgnoreString
FolderName,ou:                         attributeType.11         :CaseIgnoreString
title:                                 attributeType.12         :CaseIgnoreString
description:                           attributeType.13         :CaseIgnoreString
searchGuide:                           attributeType.14         :Guide
postalAddress:                         attributeType.16         :PostalAddress
postalCode:                            attributeType.17         :CaseIgnoreString
postOfficeBox:                         attributeType.18         :CaseIgnoreString
physicalDeliveryOfficeName:            attributeType.19         :CaseIgnoreString
telephoneNumber:                  attributeType.20         :TelephoneNumber
telexNumber:                           attributeType.21         :TelexNumber
facsimileNumber:                  attributeType.23         :FacsimileTelephoneNumber
preferredDeliveryMethod:          attributeType.28         :DeliveryMethod
presentationAddress:                   attributeType.29         :PresentationAddress
supportedApplicationContext:           attributeType.30         :OID
userPassword:                          attributeType.35         :Password
#
#defined for ASTROMAIL
#
alternateName:                         attributeType.36         :CaseIgnoreString
shellAccessLanguage:                   attributeType.42         :CaseIgnoreString
subscriptions:                         attributeType.43         :CaseIgnoreString
memberships:                           attributeType.44         :CaseIgnoreString
members:                          attributeType.45         :CaseIgnoreString
activityRecord:                        attributeType.46         :Text:File
Groups:                                attributeType.47         :CaseIgnoreString
subGroups:                        attributeType.48         :CaseIgnoreString
#
userid,uid:                            pilotAttributeType.1     :CaseIgnoreString
emailAddress:                          pilotAttributeType.3     :CaseIgnoreIA5String
manager:                          pilotAttributeType.10    :DN
alternateEmailAddress:                 pilotAttributeType.22    :Mailbox
lastModifiedTime:                 pilotAttributeType.23    :UTCTime
lastModifiedBy:                        pilotAttributeType.24    :DN
#
# QUIPU defined attributes types
```

```
#
treeStructure:                                  quipuAttributeType.1     :Schema
accessControlList,acl:                          quipuAttributeType.2     :Acl
eDBinfo:                                quipuAttributeType.3     :Edbinfo
masterDSA:                              quipuAttributeType.4     :DN
slaveDSA:                               quipuAttributeType.5     :DN
control:                                quipuAttributeType.15    :IA5String
quipuVersion:                                   quipuAttributeType.16    :IA5String
protectedPassword:                      quipuAttributeType.17    :ProtectedPassword
entrySecurityPolicy:                            quipuAttributeType.18    :SecurityPolicy
dsaDefaultSecurityPolicy:               quipuAttributeType.19    :SecurityPolicy
dsaPermittedSecurityPolicy:                     quipuAttributeType.20    :SecurityPolicy
inheritedAttribute,iattr:               quipuAttributeType.21    :InheritedAttribute
execVector:                                     quipuAttributeType.22    :PrintableString
relayDSA:                               quipuAttributeType.23    :DN
subordinateReference:                           quipuAttributeType.25    :AccessPoint
crossReference:                                 quipuAttributeType.26    :AccessPoint
nonSpecificSubordinateReference,NSSR:       quipuAttributeType.27 :AccessPoint
listenAddress:                                  quipuAttributeType.28    :PresentationAddress
cachedEDB:                              quipuAttributeType.29    :DN
#
# end of attributes
```