Network Working Group
Internet Draft: Lessons Learned from IMSP
Document: draft-newman-acap-imsp-lessons-02.txt

C. Newman
Innosoft

July 1997
Expires in six months

Lessons Learned from IMSP

Status of this memo

Introduction

IMSP (Internet Message Support Protocol) [IMSP] was designed and implemented to supply the support functions necessary for a large scale IMAP4 based infrastructure with highly mobile users. Although the protocol was successful in its mission, it was realized that a slightly different approach could achieve more for the Internet Standards community. Thus was born the idea for ACAP (Application Configuration Access Protocol) [ACAP].

This document will discuss the successes and failures of the IMSP protocol and how the IMSP experiment is influencing the design of ACAP.

1. The origin of IMSP

CMU (Carnegie Mellon University) has been running an experimental messaging system called AMS (Andrew Message System) for many years.

AMS has been extremely successful and has lead to a situation where mail, shared bboards, and newsgroups are used daily by people from all over the University, including non-technical departments. Unfortunately, AMS has two fatal flaws. It is dependant on AFS (Andrew File System) which inhibits scaling and cross-platform use, and is not standards based so that all clients have to be developed in-house.

In 1992, CMU begin working through the Internet standards process to bring the functionality of AMS to the standards community. CMU strongly supported IMAP4 [IMAP] as the core functionality, and created IMSP to supply the support functions which are needed in a message system but are not part of basic message access.

There are three major components of IMSP:

1) Storage for client configuration information.

2) Storage for user address books.

3) Mailbox distribution/replication support.

The first two components are successfully used today at a number of large sites. Experiments with the third component are ongoing.

2. Nomadic Users

Universities, Hospitals and other large sites need a message system where any PC or workstation can be used to access messages transparently. As tele-commuting and laptops become more popular, more individuals are faced with the problem of accessing their messages from more than one computer and often more than one platform. While IMAP4 [IMAP] allows users to access their message stores, it does not provide storage for address books and configuration information needed by these mobile users. IMSP fills this niche.

This need is so great that a significant number of sites have deployed IMAP4 and IMSP despite the immaturity of IMAP clients in 1995 and the experimental nature of IMSP. The IMAP4/IMSP combination allows users to move from machine to machine and get the same configuration and interface.

3. Client Configuration

The CMU IMSP server implementation provides server storage of

client configuration and also provides administrative defaults or mandatory settings for client configuration. Our experiments show this is a great success.

For example, many sites wish to control what appears in the "From:" header of outgoing mail, while other sites let the user do as they choose. The CMU IMSP server allows sites to configure either a default "From:" address, or a mandatory "From:" address based on the IMSP login name. This prevents users from accidentally sending mail with the wrong "From:" address. Administrators of large sites are quite fond of this feature.

The Simeon client from ESYS corporation stores a great deal of private configuration information on the IMSP server, in addition to common configuration options. The decision to create an IMSP options registry for common options as well as reserving parts of the name space with vendor specific prefixes appears to be sound.

Options appear to work well as they are implemented in IMSP, and are certainly not limited to messaging. ACAP should include an option registry with vendor specific prefixes, as well as administrative defaults and mandatory settings.

4. Address Books and Access Control

Almost every messaging system provides an interface for personal address books which is distinct from a public directory service. CMU's IMSP server provides an interface to multiple personal address books. It also provides rich access control on address books so they can be shared with other users. As soon as a client interface was created for these functions they both became very popular. They were so popular, in fact, that users started asking for the ability to "subscribe" to address books so they didn't have to wade though a large list.

The basic structure for IMSP address books was that each address book was made up of a list of entries, and each entry was made up of a set of (attribute, value) pairs. A set of basic attributes was defined, and others were permitted. This structure successfully provided the necessary flexibility.

Despite these successes, there are a number of problems with IMSP address books. Access becomes slow when they get large, and searching requires two round trips to the server. The original server implementation didn't allow spaces in address book entry names, but users soon demanded this flexibility. There were also requests for access control groups to improve sharing of address

books.

Finally, it became clear that there are two different models for address books in common use. The Unix/text-based model has a short alias for each entry which expands to the email address. The PC/GUI model uses common names which can be chosen from a list to use as the email address. IMSP used the common name as the primary key for address books, which makes implementation of the Unix/text-based model inefficient due to the two-round trips needed for searching.

The ACAP protocol should include address books with rich access control and a "subscription" capability. It needs to address the problems we've identified in IMSP.

5. Generalization of the Application Configuration problem

While IMSP was designed specifically for messaging applications, the options and address book functions could be quite useful in other applications. In addition, the mobility problem is not limited to messaging. Web browser bookmarks are a prime example of application configuration information which should be mobile.

Another observation was that the mailbox list features of IMSP didn't seem to fit with the address book and configuration portions, and each had different target markets. This recognition was the primary motivation to invent ACAP. ACAP specifies a basic model which can then be applied to support different applications.

6. Large Lists

The IMSP protocol model for address book entries and mailbox lists is a serious problem for large lists. It requires fetching the entire list, even when a client only has display space for the first 50. This can be very slow on low memory machines and over slow network connections.

ACAP should provide a way for clients to implement "virtual scroll bars" where they only have to fetch what needs to be displayed to the user. This means that ACAP needs rich server side searching and sorting with the ability to fetch deterministic parts of the resulting ordered list.

7. The ACAP "dataset" model

The CMU IMSP implementation ended up using the same database backend for all the lists (options, address book entries, address books, mailboxes). The server translated the function based commands for each of these lists into a common set of backend database operations.

ACAP can be a smaller and simpler protocol than IMSP if it provides data based commands rather than function based commands. The idea is to take the IMSP address book model and turn it in to a generic container which can hold options, mailboxes, access control groups or even web browser bookmarks.

Therefore the ACAP "dataset" model has the same structure as an IMSP address book: a dataset is a set of entries and each entry is a set of (attribute, value) pairs.

8. Conclusion

IMSP was a successful experiment which demonstrates the need for a configuration server. ACAP is the logical refinement of the ideas behind IMSP and is likely to become an important part of the Internet protocol suite.

9. References

[IMSP] Myers, J., "Internet Message Support Protocol", Experiment in progress, http://andrew2.andrew.cmu.edu/cyrus/rfc/imsp.html, June 1995

[IMAP] Crispin, M., "Internet Message Access Protocol - Version 4rev1", RFC 2060, University of Washington, December 1996.

[ACAP] Newman, Myers, "Application Configuration Access Protocol", Work in progress, June 1997.

10. Security Considerations

There are no known security issues in this memo.

11. Acknowledgments

Many thanks to Steve Hole and the ESYS corporation for their early client support of IMSP which was invaluable to this effort. Thanks also to Terry Gray for his insistence that IMSP was too application specific and that something more general was needed. And thanks to John Myers for his authorship of the IMSP specification and observation that everything could fit into the address book model.

12. Author's Address

Chris Newman
Innosoft International, Inc.
1050 Lakes Drive
West Covina, CA 91790

Email: chris.newman@innosoft.com