HyperText Transfer Protocolによる協調的なメールシステムの研究

4M－6

新田邦久　上原敬太郎　千葉滋　益田隆司

(株) 日立製作所 システム開発研究所　東京大学 理学部 情報科学科

1　はじめに

現在のメールシステムは、メール環境が整備されている場所からは便利に使用出来る。しかし、その環境の外からメールを操作することは難しい。そのために、いつでも、どこからでも容易にメールを操作することは出来ない。

本研究では、複数のメールサーバと複数のクライアント（メーラ）間でメールの一元管理を行なうメールマネージャを作成した。クライアントにWWWブラウザを利用するために、メールマネージャとクライアント間の通信にはHTTP[1]を使用した。WWWブラウザを利用することにより、さまざまな場所から全てのメールを操作することを可能とした。また、今後のメールシステムで必要となる、クライアント同士が協調的に動作する機能をメールマネージャに付加した。

2　現在のメールシステム

現在のメールシステムの大半は、メールをクライアントにダウンロードするため、POP[2]を使用するかNFSマウントを利用している。このようなメールシステムでは、それぞれのメールサーバに届いたメールは、一時メールサーバに保管される。これらのメールは、ユーザが未読メールの取得を要求した時、クライアントにダウンロードされる。そして、メールサーバから未読メールが削除される仕組みになっている。

この方法は、メールをクライアントにダウンロードするために、サーバのディスクを圧迫しない。しかしその反面、複数のクライアントからメールをダウンロードする場合、それぞれのクライアントにメールボックスを作成する必要がある。そして、ダウンロードしたメールの管理はユーザの責任になり、ユーザの負担が増大してしまう。

また、これらのクライアントもマシン環境に応じた細かな環境設定が必要である。そのため、環境が違うマシンを使用してメールを読むことは面倒である。その他にも、同一のクライアントから複数のメールサーバにアクセスするのは繁雑であるという問題がある。

これらの問題を解決するために.forwardによる転送やIMAPの使用が考えられる。しかし、.forwardでは複数のメールサーバに届いたメールを一つのメールサーバに集めることは可能であるが、結局クライアントにダウンロードするためにユーザの負担はほとんど変わらない。また、.forwardが使用出来ない環境もある。IMAPは、サーバ側でメールの管理を行なうために、複数のクライアントマシンを使用した場合でもユーザに負担をかけない。ただ現状では、メールを受け取るためのクライアントが一般的に普及していないため、いつでも、どこからでも容易にメールを読めるとは言えない。

3　メールマネージャ

本研究では、2節で述べた問題点を解決するためにメールマネージャを作成した。メールマネージャの特徴としては以下の2つがある。

- 複数のメールサーバと複数のクライアントの間で、メールの一元管理を行なう
- クライアントとしてWWWブラウザを使用する

メールマネージャは、登録されているメールサーバに届いた全てのメールを、メールサーバとクライアント間で管理することにより、ユーザからメール管理に関する負担を取り除く。例えばメールを読む場合、ユーザはWWWブラウザを使用してメールマネージャにメール取り出しを要求する。ユーザからメール取り出し要求を受けたメールマネージャは、ディスクからメールを取り出し、WWWブラウザで表示可能にするためにHTMLのタグをメールに追加しユーザに送信する。現在の所、ユーザがメールマネージャに要求出来る基本的な機能としては、メール取り出し、メール格納、メール送信、メール返信、メール削除、メール移動、フォルダー作成、フォルダー削除の8機能がある。メールの受信確認は30秒ごとに行ない、メールがメールサーバに到着した場合はユーザに知らせ、ユーザからメール取り込みの命令が出された時にメールマネージャのディスクに格納する。

クライアントにWWWブラウザを使用する理由は、一般的に普及しているから、また、メールマネージャのURLを指定するだけで世界各地から簡単にアクセス出来るからである。つまり、ユーザはいつでも、どこからでも簡単にメールを操作することが出来ると言える。

メールマネージャは、HTTP、POP、SMTP[3]の3つのプロトコルを使用している。HTTPはWWWブラウザとの通信を行なうために、POPはメールサーバよりメールを取り出すために、SMTPはメールを発信する場合、登録してあるメールサーバのSMTPサーバを利用するために使用する。

A Study of a Mail System Using the HyperText Transfer Protocol.

Kunihisa Nitta, Keitaro Uehara, Shigeru Chiba and Takashi Masuda.

System Development Lab, Hitachi, Ltd, Dept of Information Science, University of Tokyo.

4 協調的な機能

今後のメールシステムでは、ワークフロー制御等のクライアント同士が協調的に動作する機能が必要となる。しかし、協調的な機能を全てのクライアントに付加するのは難しい。また一部のクライアントやメールサーバに付加すると、機能を使用出来る環境が限定されてしまう。この問題を避けるために、本研究では、メールマネージャに協調作業の支援を行なう機能を付加した。クライアントは他のクライアントと協調的に作業を行ないたい場合、メールマネージャに処理を要求する。要求を受けたメールマネージャは他のクライアントとメールを送受信し協調的な動作を行ない、ユーザからの要求を処理する。これにより、メールマネージャを利用する全てのクライアントで同じ機能を使用することが出来る。現在は、以下に示す協調的な機能をメールマネージャに作成している。

- アンケート機能

  ユーザのアンケートの作成、集計を支援する機能。ユーザは、メールマネージャが提供するフォームを使用しアンケートの作成、送信を行なう。メールマネージャは、アンケート結果が全て集まるか制限時間を越えた時点で自動的に結果を集計し、アンケート関係者全てに結果を送信する。

- ワークフロー機能

  メールを指定した順序で巡回させる機能。ユーザはメールを送信する時、ワークフローの使用を指定し巡回順序を設定することにより、メールを巡回させることが出来る。

現在は上記の2機能だけであるが、今後も協調的な機能を実装し、より多くの協調作業の支援を行なう。

5 実験

上記の機能を持つメールマネージャを Sun SPARCstation 20 上に C 言語を使用してサーバの形で実装を行なった。また、比較のため HTTP サーバと CGI を使用したメールマネージャを作成した。そして、C 言語で実装したメールマネージャと CGI で作成したメールマネージャのどちらが効率がよいか調べる実験を行なった。なお、CGI は UNIX で使用されている MH を使用して作成した。

具体的な実験内容は、複数のプロセス(1個から50個)がメールマネージャに連続にアクセスし、メールを取り出した場合の処理時間の測定である。測定用メールには 1Kbyte と 100Kbyte のメールを用意した。メールの内容は、エンコードが行なわれていない ASCII テキストである。なお、測定用クライアントは Sun SPARC station 5 上で動作する。また、クライアント・サーバ間は 10Mbps のイーサネットで接続されている。CGI は Sun SPARCstation 20 上の NCSA HTTP サーバで動作する。

実験結果を図1に示す。図中の MM(1KB) と MM(100KB) は本研究で C 言語を使用し実装したメールマネージャで 1Kbyte と 100Kbyte のメールを取り出した場合の処理時間、CGI(1KB) と CGI(100KB) は CGI で 1Kbyte と 100Kbyte のメールを取り出した場合の処理時間である。実験の結果、それぞれの場合で一秒間に処理するプロセス数は、C 言語で実装したメールマネージャで 1Kbyte のメールを取り出した場合で約 14 プロセス、100Kbyte で約 5 プロセス、対して CGI は両方とも約 3 プロセスである。CGI に比べ、C 言語で実装したメールマネージャで 1Kbyte を処理する時間と 100Kbyte を処理する時間の差が大きい。これは、C 言語で実装したメールマネージャでは、メールのエンコード部分をデコードするために、メール内を全てサーチするからである。メールマネージャと CGI の速度を比較するとメールマネージャの方が優れている。CGI が遅い理由は MH を呼び出す際のオーバヘッドのためと考えられる。また CGI は nobody の属性で行なわれるために、メールを管理する場合のセキュリティに問題が発生する可能性が高い。この結果より CGI より、C 言語でサーバとして実装したメールマネージャの方が効率がよく安全であると考えられる。

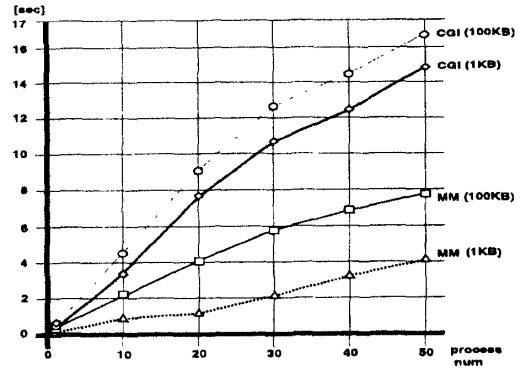

図1 メール処理時間

6 おわりに

本研究では、メールサーバとクライアント間でメールの管理を行なうメールマネージャを開発し、ユーザが WWW ブラウザを使用してメールを読むメールシステムの提案、実装、測定を行ない、メールマネージャと WWW ブラウザを使用するメールシステムの有効性を示した。

今後の課題として、4節で述べた協調的な機能の充実、共通メールボックスの実装や MIME への対応、セキュリティの強化、メールをクライアントに送信する際のメールの暗号化等があげられる。

参考文献

[1] R. Fielding, J. Gettys, J. C. Mogul, H. Frystyk, and T. Berners-Lee. "Hypertext Transfer Protocol – HTTP /1.1", December 1996.

[2] J. Myers and M. Rose. "post office protocol - version 3", RFC1939, May 1996.

[3] J. B. Postel. "SIMPLE MAIL TRANSFER PROTOCOL", RFC821, August 1982.