研究論文

メイリングリストの冗長データ軽減システムの設計と評価

広瀬　雄二

概　要

メイリングリストは導入が容易で送信者に特別な知識を要求しないものであるため、グループ間の情報共有をする上での有用な伝達手段であり続けている。そのいっぽう、1通の送信メイルを加入メンバー全員に配送するという性質上、バイトサイズの大きなデータを送付した場合、それに宛先数を乗じた転送量が発生するため、回線やメイルサーバの圧迫を容易にもたらす。また、必ずしも全員が保持する必要のないデータであっても、全員に配送され全員の計算機内に保存され、無駄な計算機資源消費を生むこともある。

技術的には大容量データの配付にはそれに適した手段を用いるのが望ましいが、実際には手段変更が困難な状況も多い。本稿では、利用者の手順を変えさせることなく、メイリングリスト利用時の諸問題を回避するシステムa2e を提案し、その効果を検証する。

Abstract

A traditional tool called ‘mailing list’ is still in use because of its installation simplicity for administrators and easiness of use for users. Meanwhile, since mailing list amplifies a message by the number of recipients, network traffic floods and excessive load escalation of mail servers are easily caused by only ‘one’ huge message. In addition, some of attached data are unnecessary for certain part of members.

Although it is preferred to use other tools than mailing list to interchange large files among members, there are some cases that it is difficult for users to change their way of daily tasks. In this paper, we propose a system that avoids problems around mailing list and evaluate it.

1　背景

多くの局面で電子メイル[1]が用いられるようになった。MIME の普及によって文字メッセージのみならず､バイナリデータがいわゆる「添付ファイル」の形で送信できるようになり、ファイル転送手段としても電子メイルが利用されるようになった。通信回線の高速化とメイルサーバの性能向上によりMB(メガバイト)単位のバイナリデータを電子メイルで送付しても、個人間の授受では遅延を意識することなく利用できるようになった。

しかし送付先がメイリングリストである場合は、依然送信メッセージの転送量への配慮を送信者がする必要がある。メイリングリストは1通のメイルを加入メンバーの数に増幅するものであるため、たとえ1MBのメッセージでも100人に送れば転送量は100MBになり、容易にメイルサーバや回線の圧迫をもたらす。

このため、一般的にはグループ間のファイル授受には他のツールを使うことが望ましいとされる。しかし、既にメイリングリストを日常業務に利用している場所では、全員の利用形態の変更が容易でない場合もある。

本稿ではMIME[2]形式の電子メイルに添付されたバイナリデータを、外部のデータ配付用Webサーバに自動的に保存し、メイリングリストに送付するメッセージは本文とWebサーバに保存した添付ファイルへのリンクだけに置き換えることで、利用者の使い勝手を変えることなくメイリングリスト使用時に発生しがちな諸問題を解決するためのシステムを提案し、その設計・構築および運用した結果について論ずる。

2　ファイル添付の問題点

バイナリデータを添付した同一のMIMEメッセージをメイリングリスト経由で多人数に配送する場合の問題点を挙げる。

2.1　大きさ

添付されるバイナリデータは文書を保持する形式であることが多い。文章の

```
pan{ta01002}% ls-l ex*
-rw-r--r--  1 ta01002 32547 ex2.b64（Base64 符号化したデータ）
-rw-rw-r--  1 ta01002 24064 ex2.doc（Word 文書符号化したデータ）
-rw-rw-r--  1 ta01002    26 ex2.txt（元のテキストデータ）
```

図1：同一文章の異なる符号化でのファイルサイズ

みで表現されうる情報はプレインテキスト[1]で表現可能だが、これをワードプロセッサ固有の書式で符号化した場合、元の文章に含まれる文字数以上のバイト列が付加されるため、必然的にデータの大きさが増す。

さらに、MIMEメッセージに添付されたバイナリデータは、通常7ビット文字集合に符号化された状態で転送され、そのままの形式で受信者の元に届く。7ビットあるいは、それより小さいビット長で表現できる集合を8ビット幅のバイト列に載せることから、符号化されたデータは元のバイナリデータよりも大きくなる。たとえば、様々なMUA[2]で符号化に用いるBase64[3]の場合、3バイト＝24ビット＝8ビット×3のデータ並びを、ASCII文字だけで表現できる6ビット×4の4バイトの並びに変換するため、理論上データ量は元のバイナリデータの4/3倍になる。

そのようなメッセージをメイル送信する場合、1対1の授受では無視できる程度の大きさでも、メイリングリストで送信された場合、転送量増加は配送宛先数を乗じた値となる。大きな添付ファイルをメイリングリストに流すことが忌避されるのはこのためである。

2.2　冗長性

添付されたバイナリデータそのものにある冗長性がもたらす問題点には、以下のものがある。

2.2.1　データ容量の問題

添付されたバイナリデータを受信者が参照する場合、受信者の計算機で

[1] 文字コードのみの羅列からなる、文章を記したデータ。
[2] Mail User Agent；電子メイルの読み書きをするためのユーザ向けソフトウェア。

1. 7ビット文字集合で表現されたデータを8ビットデータに復元する
2. 復元された8ビットデータを一時ファイルに保存する
3. 一時ファイルを専用アプリケーションで開く

という手順を経て中味が参照できる状態になる。つまり、同じ内容を表すデータがメイル本文と一時ファイルの最低二箇所に存在することになる。

また、メイリングリストで複数人が添付データを受け取った場合、装飾と符号化で肥大化したデータが、受信者数だけコピーされることになるため、消費されるハードディスク容量は元のデータ量に受信者数を乗じた大きさになる。

たとえば、以下の簡単な文章(26バイト)を文書データ化して送信する場合を考える。

タイトル

これは本文です。

これをMicrosoft Word 2003(以下Word)で保存すると約24000バイトになる[3]。これをBase64で符号化したものは約32000バイトとなる(図1)。このケースで仮に100人のメイリングリストにメッセージを送る場合、プレインテキストなら26B×100＝26000B＝25.4KB、Word文書なら32547B×100＝3178.4KBが合計転送量となる。

2.2.2 旧版保持の冗長性に起因する問題

いったん送信した文書を訂正する場合、訂正前の文書と訂正後の文書が両方受信者の手元に残る冗長性もしばしば問題を引き起こす。文書の修正履歴が重要な意味を持つものであれば、受信者が古い版を保持することにも意味があるが、そうではなく最新情報のみ入手できればよい性質の文書配付の場合、受信者の手元に新版、旧版があったときに誤って旧版を利用する可能性がある。

たとえば申請書類のような文書は、紙ベースの処理であれば申請窓口で配付している申請用紙を、提出の直前に入手して記入する。手元に古い申請書があったとしても、それがいつまで有効なのか考えるより窓口のものを利用する方が

[3] 実際のバイトサイズはバージョンやシステム環境など条件によって異なるが、ここではプレインテキストより大きくなることを論ずる。

面倒がないと考えるのは自然なことである。そのような、最新版であることが意味を持つ書類を配付する場合、メイリングリスト送付のように旧版が蓄積されていく形態の配付方法は、データ記憶領域の増大を招くという問題だけではなく、ファイルの新旧の一覧性を欠くため、古い情報を誤って利用する危険性もある。

2.3 汎用性と永続性の問題

バイナリデータは、なんらかの情報を特定の規則で符号化したものである。符号化規則は個人、あるいは特定の組織・団体で策定するもので、その使用を公開するか否かは策定者の都合次第で決まる。商用ソフトウェアのファイル保存形式は商業的な都合で非公開である場合がある。そのような場合、保存したデータは使用したアプリケーションプログラムに依存することになる。ソフトウェアの種類や、バージョンに依存する度合が高ければ高いほど、誰でもどこでも内容を取り出せる度合(汎用性)と長期的将来に渡って取り出せる可能性(永続性)が低くなる。

商用ソフトウェアの場合は、いわゆる「バージョンアップ」という形で定期的にアプリケーションプログラムの変更を迫られることがある。これは定期的に人数分のソフトウェアライセンス料負担を強いられるばかりでなく、ときにはバージョン間差異によって文書の表現結果にずれが生ずることもある。またそもそも、特定の商用アプリケーションプログラムに依存するバイナリデータは、同じプログラムを購入していない人間には扱うことができないという問題もある。

2.4 その他の問題

メイリングリスト経由で受け取ったメッセージ群は、ファイルの形で受信者の計算機に保存される。このファイルの管理は計算機使用者である受信者本人の責任において行なわれることになる。

たとえば、このファイル群から必要な情報を検索できるかは、受信者の利用しているソフトウェアに効率的な検索機能が備わり、なおかつ受信者がそれを把握しているかにかかっている。メイリングリストの構成員が全て均質のシス

テムを利用しているとは必ずしもいえないし、全員が高い利用技能を持っているという仮定も置けない。検索できる条件が揃っている受信者でも、機械故障などを原因とするファイル損失により必要な情報を取り出せなくなる危険性がある。

以上のように、配送したメッセージ群の中から必要な情報を取り出せることの条件を受信者側の負担としている点は、局所的な情報格差を発生させる原因となりうる。

3 a2eシステム

前節で述べたいくつかの問題点を解決する方策について考察し、その一案を示し、その実装について説明する。

3.1 冗長性問題の解決

サイズの大きなファイルはメイリングリストには直接流さず、Webスペースなどの誰でも随時アクセス可能な場所に配備し、メイリングリストにはその場所情報のみ送付する、つまり外部参照形式で通知するというのが一般的な解だが、現実的にはそのルールを構成員全員に適用するのは困難である。したがって、バイナリデータが添付されたメッセージを送るという送信者側の手順を変えることなく、添付データを自動的に外部参照形式に変換し、データの場所のみを全受信者に送付すれば冗長性排除の目的は達成できる(図2)。

3.2 汎用性・永続性問題の解決

前項同様、送信者の手順を強制的に変えさせることなく、特定アプリケーションプログラムに依存したバイナリデータを汎用的な形式に自動的に変える方法を取ればよい。具体的には、Microsoft Word、Microsoft Excelで作成した文書を、テキストファイルに変換したものをWebスペースに配備し、その場所情報も送付メイルに付加することで、専用アプリケーションプログラムがない場合でも内容にアクセスできるようにする(図2の「TEXT」へのリンク部分)。

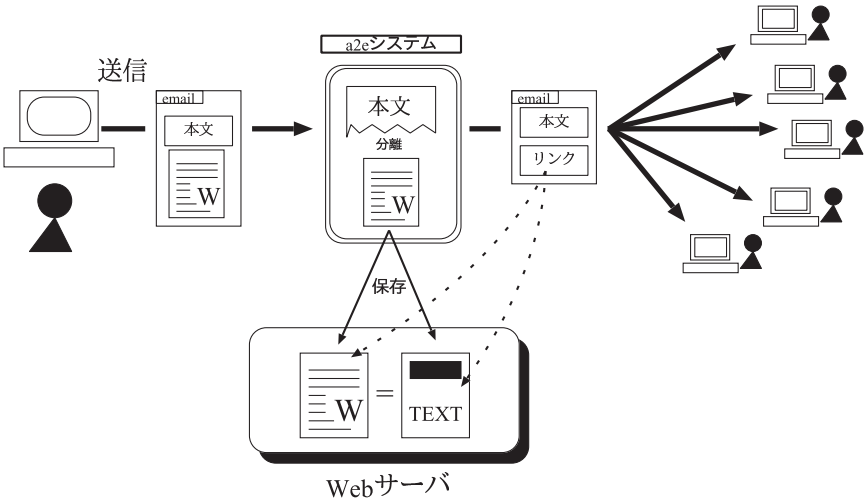

図2：動作概念図

3.3 実装環境

図2に示したように、添付形式のものを外部参照形式に変換して再配送するシステムatta2ext（Attachment to External；以下a2e）を設計・実装した。

実装はRuby[4]を用いて行なった。a2eが利用しているもの、前提としている環境について簡単に示す。

Ruby プログラミング言語Ruby。本システムはRuby1.8.7で実装・検証を行なった。

tmail RubyでRFC2822メッセージ[4]を処理するためのライブラリ。Multipartメッセージの分解と、外部参照形式への再構築操作、送信操作で利用した[5]。

Hyper Estraier 様々な文書書式に対応した全文検索システム。メイリングリストに送付されたメッセージの、本文・添付されたバイナリデータ全てを検索インデックス化し、かつ、Webインタフェースでフレーズ検索する機能を供給する[6]。

[4] いわゆる電子メイルの書式。現在は RFC5322 に更新されている。

表1：処理の有無によるメッセージ総容量比較

種　　別	総　容　量
元メッセージ	51228KB
a2e変換後メッセージ	902KB

wvWare　Microsoft Wordの文書ファイルを解読し、プレインテキストやHTMLなどの可読形式に変換するライブラリ[7]。

xlhtml　Microsoft Excelの文書ファイルを解読し、プレインテキストやHTMLなどの可読形式に変換するコマンド[8]。

pdftotext　PDF内のテキストを抽出するユーティリティプログラム。Poppler[9]パッケージに含まれる。

Apache　Webスペースに保存したメッセージの本文と添付データを発信し、全文検索プログラムのCGI基盤を供給するWeb サービスデーモンプログラム[10]。

qmail　電子メイル配送エージェントプログラム。メイリングリストへの送信メッセージの内容を標準入力としてa2eシステムを起動する。a2eではqmailによる配送であることを仮定[5]して送信者アドレス(sender)と受信者アドレス(rcpt)の抽出を行なっている[11]。

4　実験結果

作成したa2eシステムを、実際に運用されているメイリングリストに適用した結果を示す。

4.1　実験環境

東北公益文科大学で運用されている教員用メイリングリスト宛に送信されたメッセージをフックして、Webアーカイブの蓄積と添付データ除去後のメイル送付を行なった。

[5]Postfix（http://www.postfix.org/）による配送も認識する。

表2：元メッセージの添付データ有無による分類

種　別	総数	総バイト数	平均バイト数
添付データなし	143	298295	2086B
添付データあり	79	51358566	650108B（≒635KB）

4.2　冗長性軽減効果

a2eシステム設置は2010年7月20日に行ない、本稿執筆現在も稼動し続けている。そのうち、2010年7月21日から2010年10月4日までの全222通について集計した。

・送信者による元メッセージ

・a2eを通して添付データを除去し外部参照形式に差し替えたメッセージ

それぞれを集約したフォルダ(2KBバイトブロック)の容量は表1のとおりであった。これによりa2eによる変換によって受信者のこの期間のディスク使用量が1/50以下になったことが分かった。

該当期間の通過メッセージの内訳を示したものが表2である。添付データを含むメッセージの割合が総数に占める半分以下でありながら占有バイト数が本文のみのメイル総バイト数の約170倍にもなっていることが分かる。

4.3　データの汎用化と検索機能提供効果

元メッセージの添付部分は、a2eによって以下のような外部参照情報に置換される。

```
以下のファイルが添付されました。
----------
添付ファイル：添付されたファイル名
→添付ファイル名の外部参照URL
同じファイルをASCII文字のみのファイル名で保存した外部参照URL
----------
この日の送付文書一覧は次のURLにあります。
同じ日付けのメッセージ一覧と検索用WebのURL
```

網掛けで示した部分は、実際にはそのときの日付けやファイル名に応じて自動生成されたWebデータのURLとなる。受信者がGUIベースのMUAで各URL

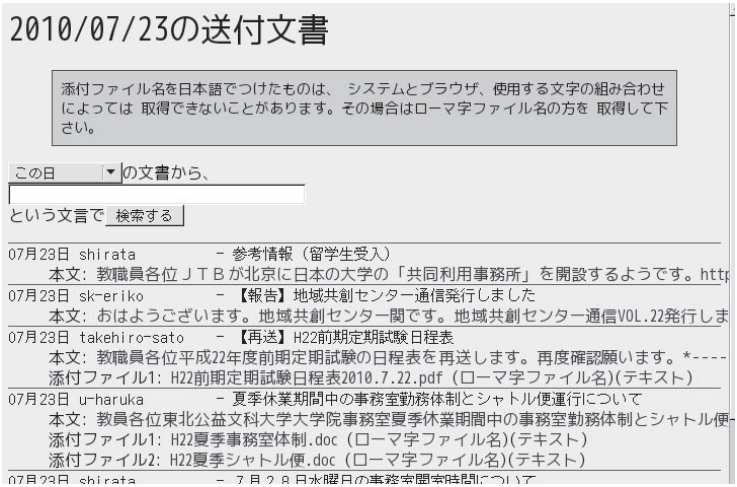

図3：日ごとの文書一覧と検索フォーム

を見た場合、その部分をクリックするなどすると実際のデータにアクセスできる。この操作はメッセージに添付されたファイルを開くのとほぼ同じ操作である。また、末尾にある検索用WebのURLをブラウザで開くと図3のようなWebページが現れ、一覧にあるメッセージ本文や添付ファイル名をクリックすると、該当する中味にアクセスすることができるようにしてある。添付されたバイナリデータが文書ファイルである場合は、ファイル中の本文のみを抽出したテキストファイルの形式も保存してあるので、永続的な「内容への到達可能性」が確保される。

また､このページにある検索窓に検索フレーズを入力し､検索範囲を「この日、この月、この月と前月、最近1年分、1週間以内～ 10週間以内」のいずれかに指定すると、指定した期間に送付された全文書、全添付ファイル[6]から、検索フレーズを含むファイルへのリンク一覧が表示される。これにより受信者のメイル購読環境に左右されることなく、メイリングリストに投稿された情報を的確に引き出すことができる。

[6] 現状では Word，Excel，PDF ファイルに対応している。理論的にはテキスト抽出フィルタを用意できる文書形式であればどんな形式でも検索対象に入れることができる（hyper estraier の性質)。

また、Webスペースのファイルシステムのバックアップを取りデータ損失に備える運用を行なうことで、受信者側が各自でメッセージフォルダのファイル損失予防策を取る必要がなくなる。

5　結論

本稿で提案・設計・実装したa2eシステムを、メイリングリストの前段フックとして利用することで、メイリングリストにバイナリデータを添付して送る場合の、冗長性、高資源消費、データの低汎用性といった問題が回避でき、なおかつ時系列を指定できる検索機能によってメイリングリストに投稿された情報の長期的活用を促すことにもつながることが確認できた。

今後は、実際にメンバー全員で利用する場合の現実的諸問題の発見に取り組み、メイリングリストを核とした、より円滑で効果的な情報共有システムを構築する礎としたい。

6　おわりに

本稿で実装したa2eシステムは

`http://www.gentei.org/~yuuji/software/a2e/atta2ext.rb`

で入手できる。`Revision 319:44016c452301 2010-07-20 09:17+0900`の時点で約500行とコンパクトに実装できたのはRubyと各ライブラリに負うところが大きい。それぞれの開発者陣に敬意を表する。

参考文献

[1] Simple Mail Transfer Protocol; Network Working Group, Request for Comments RFC2821, April 2001

[2] Multipurpose Internet Mail Extensions（MIME）Part Four : Registration Procedures; Network Working Group, Request for Comments - RFC2048, November 1996

[3] The Base16, Base32, and Base64 Data Encodings; NetworkWorking Group,

Request for Comments - RFC3548, July 2003
［4］ オブジェクト指向スクリプト言語Ruby; Ruby コミュニティ, まつもとゆきひろ; http://www.ruby-lang.org/（ref.2010-10-08）
［5］ TMail - A Ruby Email Handler; TMail Project, AOKI Minero, et al.; http://tmail.rubyforge.org/（ref.2010-10-08）
［6］ 全文検索システムHyper Estraier; 平林幹雄; http://fallabs.com/hyperestraier/ Last Update: Tue, 06 Mar 2007 12:05:18 +0900;
［7］ wvWare, library for converting Word documents; Dom Lachowicz, Caoln McNamara; http://wvware.sourceforge.net/（ref.2010-10-08）
［8］ The xlHtml Homepage; Charles Wyble; http://chicago.sourceforge.net/xlhtml/（ref.2010-10-08）
［9］ Poppler; Albert Astals Cid; http://poppler.freedesktop.org/（ref.2010-10-08）
［10］ The Apache HTTP Server Project; The Apache Software Foundation; http://httpd.apache.org/（ref.2010-10-08）
［11］ The qmail home page; The Apache Software Foundation; http://www.qmail.org/top.html, Last modified: Sat May 15 11:39:33 EDT 2010