東京家政学院筑波女子大学紀要第４集　163～177ページ　　2000年

＜研究ノート＞

電子メール自動集計処理システムⅠ
－ヘッダーの処理－

江戸　浩幸・坂本　義行

Automatic Processing System on Internet Text Messages

Hiroyuki EDO abd Yoshiyuki SAKAMOTO

概　要

本研究の目的は、世界的に普及してきたインターネットを介して情報の発信、受信を行う電子メールシステムを処理対象とする。電子メールシステムは、一定の基準にしたがってさまざまな情報が発信、受信され、メールボックスに記憶される。本システムは、利用者が、このメールボックス内にあるメール情報の中から必要な情報だけを取り出し、検索や集計処理が出来るシステムを作成することを目的とする。ヘッダー内の情報を検索する為に、文字列パターンの処理を行うプログラムを作成した。このプログラムの利用者は、システムを汎用的な環境下で、簡単な操作により必要な情報のみを取り出すことができる。その処理方法、プログラミング、実行方法、結果について説明する。

キーワード：電子メール、メールヘッダー、ファイル処理、文字列処理、表作成

１　処理の対象

メールボックス内のメール情報を１ファイルと定義する。１ファイルはｎ通のメールからなり、１通のメールを１メールとする。

１メールはヘッダーと本文の２つの部分から構成されている。1メールは、文字列「Return-Path:」から、つぎの文字列「Return-Path:」の前までとする。ヘッダー部分にはメールが誰から、いつ届いたかなどの情報が「RFC822: Standard for ARPA Internet Text Messages」の規格に従って記述されている。本文部分にはメッセージ本体（メール本文）が記述されている。これら２つの部分は空行によ

って互いに分割されていている。ただし、空行とは改行コードが連続している部分をいう。

今回の集計システムでは、メールのヘッダー部を抽出処理の対象とした。ヘッダー部内の「差出人アドレス」、「発信日付」、「表題」の各項に記述されている文字列を抽出する。

「RFC822:ARPAインターネットのための基準」[1),2),3)]

RCFとはRequests for Commentsの略である。これは、インターネットがどのように動作するのか、どのように利用するのか、今後どのようになるのかについて記述されている。

文書はすべて通し番号で区別され、1200種類以上ある。「RFC822」は「Standard for ARPA Internet Text Messages」というタイトルが付けられていて、１メールのヘッダー部分と本文部分の分割、及びヘッダー部分の文法と順序を定義している。

１メールの形式を例によって表示する（図１参照）

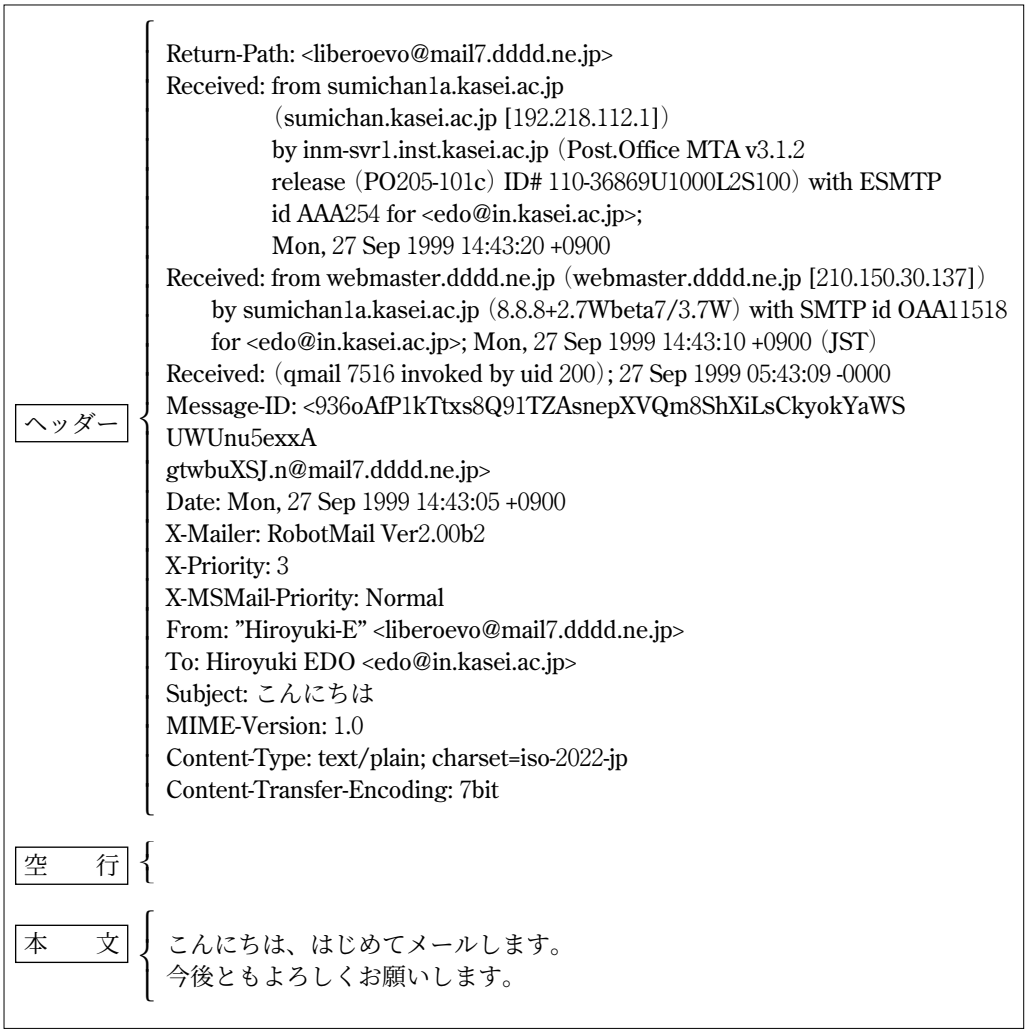


図１　１メールの形式例

2　ヘッダー部の各項目の形式

2.1　差出人アドレスの項

差出人アドレスの項には、文字列「From:」を先頭に差出人のメールアドレスが必ず記述されている。その他に差出人の名前やニックネーム等が付記される。終端は改行コードで終わる。以下「From項」とする。

メールアドレス

手紙を間違いなく届けるために名前と住所を記述する必要があるように、電子メールをきちんと送受信するために必要なのがメールアドレスである。

メールアドレスは、名前と住所を合わせたようなものである。エアメールを書く場合に、名前、番地…国名という順に記述するように、メールアドレスも、先頭に相手のコンピュータに登録されているユーザー名、つぎに「@」アットマーク、ドメイン名（組織名）、組織の種別、国別コードが、「.」ピリオドで区切られ、半角英数字で記述されている。

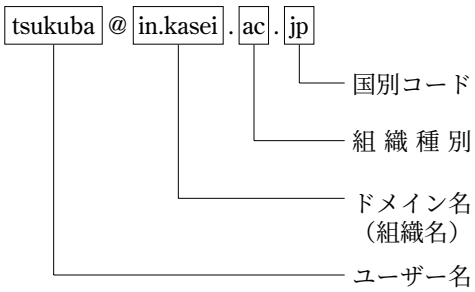


図2　電子メールアドレスの書式例

形式の例

1. From: hanako tsukuba <tsukuba@in.kasei.ac.jp>
2. From: ”筑波　花子” <tsukuba@in.kasei.ac.jp>
3. From: ibaraki@in.kasei.ac.jp（kenji ibaraki）
4. From: ibaraki@in.kasei.ac.jp（茨城 ケンジ）
5. From: aduma@in.kasei.ac.jp

2.2　発信日付の項

発信日付の項には、文字列「Date:」を先頭に差出人がメールを発信した日付が自動的に記述されてる。日付は、曜日、日、月、年、時間、時差又は標準時（以下、標準時とする）の順にスペースで区切られている。終端は改行コードで終わる。以下「Date項」とする。

形式の例

1. Date: Mon, 10 May 1999 10:10:10 +0900
2. Date: Sat, 28 Aug 1999 20:34:29 +0200
3. Date: Tue, 20 Mar 1999 14:50:28 EST
4. Date: Fri, 12 Mar 1999 10:28:23 PST

「+0900」や「+0200」はグリニッジ標準時との時差を表わす。「EST」や「PST」はグリニッジ標準時を基準に地域･国によって用いられる標準時刻を表わしている。

形式の例

1. GMT: Greenwich Mean Time
   グリニッジ標準時
2. ＥＳＴ: Eastern Standard Time
   米国の東部標準時、グリニッジ標準時より５時間遅れ
3. ＰＳＴ: Pacific Standard Time
   米国の太平洋標準時、グリニッジ標準時より８時間遅れ

2.3　標題の項

標題の項には、文字列「Subject:」を先頭にメール本文の内容を表わす標題が記述されてる。ただし、空項も可とする。終端は改行コードで終わる。以下「Subject項」とする。

形式の例

1. Subject: こんには
2. Subject: Re: こんにちは

ただし、「Re:」は、返信「Reply」を行ったことを示していて、メールソフトが自動的に記述してくれる。

３　処理手順

３.１　From項の処理

文字列「From:」につづく文字列の中から、メールアドレスだけを抽出する。

３.１.１　山括弧「<>」でメールアドレスが囲まれている場合

形式の例

From: hanako tsukuba <tsukuba@in.kasei.ac.jp>

開山括弧「<」を区切り文字として差出人の名前とメールアドレスとを分割する。次にメールアドレスの終わりにある閉山括弧「>」を区切り文字としてメールアドレスだけを抽出する。

３.１.２　丸括弧「( )」で差出人の名前やニックネームが囲まれている場合

形式の例

From: ibaraki@in.kasei.ac.jp（茨城 けんじ）

丸括弧「( )」とその中にある差出人の名前やニックネームを削除してメールアドレスを抽出する。

３.１.３　差出人の名前やニックネームが付記されていない場合

形式の例

From: aduma@in.kasei.ac.jp

そのまま、メールアドレスとして抽出する。

３.２　Date項の処理

形式の例

Date: Tue, 20 Mar 1999 14:50:28 +0900

文字列「Date:」につづく文字列の配列を並べ替え「日 月 年 時間」、「曜日」、「標準時」の三項目に分けて抽出する。

３.３　Subject:の処理

形式の例

Subject: こんには

文字列「Subject:」に続く文字列を抽出する。

４　プログラミング

本章では、前章で述べた各項目の判定データ(文字列)の抽出、並びに集計の各処理を行うプログラムについて述べる。

４.１　プログラミング言語の選定

基本システムを汎用的に利用可能な環境に置くことが第一であり、処理に最適な言語を選択することが次に重要である。処理を大きく分けると、各項目の判定及び抽出と,表形式での集計に分かれる。

文字列の判定、抽出の処理に適し、汎用性を備えたプログラミング言語としては、LISP、AWK、Perl等が考えられる。いづれも記号処理言語である。

AWKは、正規表現を使った文字列検索機能が強力でMS-DOS、UNIX上で稼動し、本学でも利用可能な環境にある。したがって、AWKをプログラミング言語として採用した。

AWKの特徴[4),5)]

AWKの名前の由来は、3人の開発者、Alfred V.Aho、Peter J.Weinberger、Brian W.Kernighanの頭文字を取ったものである。

この言語は、入力パターンに応じていろいろなアクションを実行することによって、データ処理からプログラムの試作品作りまでの様々な仕事を処理できるようにしたファイル処理言語である。AWKプログラムは、入力文字列の中から見つけるべきパターンと、見つかったときに実行するアクションの組合せからなっている。AWKは、それらのパターンのどれかと適合する部分をファイルの中から探し出し、一旦そのような部分が見つかれば対応するアクションを実行する。

パターンは、正規表現、文字列、数、欄、変数などを比較演算と組合せることにより入力行を検索できる。アクションは、一般の言語でのプログラムに相当する。パターンで選択されたアクションは、その部分に記述されて

いる出力や計算などの処理を行う。このアクションを記述する言語はC言語に似ているが、宣言を持たなかったり、文字列や数も組み込みデータ型として備えている点が異なる。

AWKは入力ファイルを行単位で読込み、それぞれの入力行を自動的に欄に分解する。入力、欄への分解、記憶領域の管理、初期化など、自動的に行われるのでプログラムは通常、他のプログラミング言語よりもはるかに小さなプログラムとなる。AWKにはいくつかのバリエーションがあるが、今回は「GUN AWK」（通称GAWK）を日本語化したJGAWKを使用した。

AWKプログラムは、表計算の処理に適した形式で出力が可能である。その表計算には、表形式の集計に適し、普及度の高い汎用ソフトの「Microsoft-Excel」を使用した。

「Microsoft-Excel」は、「Microsoft-Windows」環境下で稼動するアプリケーションソフトの一つで、一般に「表計算（スプッレッドシート）ソフト」と呼ばれている。

Excelの特徴

表計算（スプッレッドシート）ソフトとは、集計表や一覧表のように、データを種類ごとに規則的に並べた表形式のデータの集計や計算などの処理を行う汎用ソフトをいう。スプッレッドシート（Spread Sheet）の画面は、縦の列と横の行から構成されていて、そこに出来る1つ1つの桝目をセル（cell）といい、表全体をワークシートという。

特徴として

1. セルは数値や文字列　計算式を入力して表の作成や自動計算を行う
2. セルに入力されたデータを変更すると関係する計算式によって自動的に再計算を行う
3. 表のデータを使って　各種形式のグラフを作成する
4. 表の中のデータを並べ替えるソート　必要なデータを抽出することが出来る

等の機能を備えている。

5　プログラム

今回の電子メール検索プログラムの名称を「mail.awk」として図３に示す。

次節から、プログラムの動きを順を追って説明する。

5.1　パターンBEGINとENDの処理

AWKはファイル処理言語であるから、読込むファイルの最初のレコード（最初の行）の前と最後のレコードの後の処理が必要である。この処理をBEGIN と END パターンが行う。

```
BEGIN{
        print ”MailAddress,” ”Date, ” ”Week,”
        ”StandardTime,” ”Subject” ;
      }
```

パターン「BEGIN」に伴うアクションは、入力文字列が読込まれる前に実行される。アクション「{文１}」は、抽出される項目の見出しを１行目に出力する。このアクションが実行されると「MailAddress」：メールアドレス、「Date」：日付、「Week」：曜日、「StandardTime」：標準時、「Subject」：標題の各項名をカンマで区切り出力する。

```
END{ }
```

「{ }」内が空であるから、ここではとくに処理は行われない。

```
BEGIN {
        print ”MailAddress,” ”Date,” ”Week,” ”StandardTime,” ”Subject”;
       }
/^Return-Path: /{ mode=0; from=” ” ; date=” ” ; subject=” ” ;}
/^From:/{ from=address($0); }
/^Date:/{ date=day($0); }
/^Subject:/{ subject=splithead($0); }
/^$/  {
    if(mode==0) {
                 mode=1;
                 print from”, ”date”, ”subject;
                }
    }
END {
    }
function address(ptr) {
                      addr1=splithead($0);
                      n=split(addr1,addr2,”<”);
              if(n>1) {
                      n=split(addr2[2],addr3,”>”);
                      addr=addr3[1];
                     }
                 else{
                      addr=addr1;
                     }
        gsub(/ ” ”/, ” ” ,addr);
        gsub(/¥([^( )]*¥)/,” ”,addr);
        gsub(/¥([^( )]*¥)/,” ”,addr);
        return addr;
                     }
function day(ptr) {
                   addr1=splithead($0);
                   split(addr1,addr2,” ”);
                   addr=addr2[2]” ”addr2[3]” ”addr2[4]” ”addr2[5]”,”addr2[1] ” ’ ”addr2[6];
                   return addr;
                  }
function splithead(ptr) {
                         m=index(ptr,”: ”)+2;
                         addr1=substr(ptr,m);
                         return addr1;
                        }
```

図３　電子メール検索プログラム「mail.awk」

5.2 文字列照合パターン

「/」スラッシュで囲まれた文字列パターンを照合する。

5.2.1 ヘッダー先頭の判定

```
/^Return-Path:/{ mode=0; from=" "; date=" ";
subject=" ";}
```

レコードの先頭が「Return-Path:」と一致する文字列を検索すると、アクション「{文２}」を実行する。文２は、変数「mode」に０(零)を代入して、先頭が「Return-Path:」と一致する文字列をヘッダーの先頭と判定し、「from」、「date」、「subject」の各変数の値を０(零)にリセットする。

5.2.2 From項の判定

```
/^From:/{ from=address($0); }
```

レコードの先頭が「From:」と一致する文字列を検出すると、アクション「{文３}」を実行する。「address」は定義された関数（後述）である。

5.2.3 Date項の判定

```
/^Date:/{ date=day($0); }
```

レコードの先頭が「Date:」と一致する文字列を検出すると、アクション「{文４}」を実行する。「day」は定義された関数（後述）である。

5.2.4 Subject項の判定

```
/^Subject:/{ subject=splithead($0); }
```

レコードの先頭が「Subject:」と一致する文字列を検出すると、アクション「{文５}」を実行する。「splithead」は定義された関数（後述）である。

5.2.5 ヘッダー終端の判定と出力

```
/^$/{
    if(mode==0){
                mode=1;
                print from", "date", "subject;
               }
    }
```

レコードに「$」空行を検出すると、アクション「{文６}」を実行する。

文６は、「if」文の式「mode==0」により、変数「mode」が０(零)ならば、変数「mode」に１を代入して空行がヘッダー終端と判定し、前節までで判定された変数の値を「, 」カンマで区切り、「from」、「date」、「subject」の順に出力する。変数「mode」に１が代入されたことにより、本文が空行であってもこの処理は実行されない。

5.3 関数の定義

AWKプログラムでは、ユーザーが繰り返し使用する処理を定義関数として定義出来る。これにより同じ処理を何度もスクリプト中に記述せずに済む。

定義関数は、

```
function 関数名（仮引数の並び）{
                関数本体
                ...
return　式
                                    }
```

の形で記述する。「return」の後の式が定義関数を実行した後の戻値となる。

5.4 From項の照合と抽出

From項の中からメールアドレスを抽出するために以下の関数「address」を定義する。

定義関数「address」

```
function address(ptr) {
                    addr1=splithead($0);
                    n=split(addr1,addr2,"<");
            if(n>1) {
```

```
        n=split(addr2[2],addr3,">");
        addr=addr3[1];
        }
    else {
        addr=addr1;
        }
gsub(/" "/," ",addr);
gsub(/¥([^()]*¥)/," ",addr);
gsub(/¥([^()]*¥)/," ",addr);
return addr;
        }
```

５.４.１　抽出処理対象範囲の選択

「addr1=splithead($0)」でFrom項の抽出処理対象範囲を選択する。「splithead」は定義関数（後述）である。

例 From: hanako tsukuba <tsukuba@in.kasei.ac.jp>

この範囲を「addr1」の値とする

５.４.２　文字列「<」(開山括弧) の照合とaddr1の分割

文字列操作関数「n=split(addr1,addr2,"<")」は、5.4.1項で選択された文字列「addr1」を区切り文字に指定した開山括弧「<」に従って文字列「addr2」に分割する。

結果、addr2[1]=hanako tsukubaとaddr2[2]=tsukuba@in.kasei.ac.jp>に分割され、nの値として２が与えられる。

区切り文字に指定した開山括弧「<」が文字列「addr1」に存在しない場合はnの値は０(零）となる。

５.４.３　nの値による文字列操作関数の実行制御文

```
if(n>1) {
        n=split(addr2[2],addr3,">");
        addr=addr3[1];
  } else {
        addr=addr1;
       }
```

は、始めに「if」文の式「n>1」が評価され式が真の場合､続く文が実行され、式が偽なら「else」に続く文が実行される。

５.４.４　閉山括弧「>」の参照とメールアドレスの抽出

5.4.3項で、nの値が２のとき、文字列操作関数「n=split(addr2[2],addr3,">")が 実行され、文字列addr2[2]の値（例 tsukuba@in.kasei.ac.jp>）から閉山括弧「>」を分離し、残りの文字列（例 tsukuba@in.kasei.ac.jp）をaddr3[1]とする。addr=addr3[1]として、メールアドレスが文字列「addr」の値になる。

5.4.3項で、nの値が０(零）のとき、「addr」の値は5.4.1項で選択された文字列「addr1」となる。これはメールアドレスが山括弧「< >」で囲まれていない場合である。

例

ibaraki@in.kasei.ac.jp（茨城 けんじ)、
aduma@in.kasei.ac.jp

５.４.５　メールアドレス以外の文字列の処理

文字列「addr」をメールアドレスのみにするために、以下の文字列操作関数を定義する。

```
gsub(/" "/," ",addr);
gsub(/¥([^()]*¥)/," ",addr);
gsub(/¥([^()]*¥)/," ",addr);
```

文字列操作関数「gsub(/" "/," ",addr)」は、文字列「addr」の中のスペースを削除する。続く文字列操作関数「gsub(/¥([^( )]*¥)/," ", addr)」は、文字列「addr」の中の丸括弧「( )」とその中の文字列「例（茨城　けんじ)」を削除する。

同じ処理を２度行うのは「例（suzuki(satou) kazuko)」などの二重括弧を削除するためである。

５.４.６　定義関数の戻値

定義関数の戻値を返す「return addr」によって定義関数「address」の値は抽出されたメー

ルアドレスとなり、5.2.2項の変数「from」の値はメールアドレスとなる。

5.5 Date項の照合と抽出

Date項の中から日付を抽出するために以下の関数「day」を定義する。

定義関数「day」

```
function day(ptr) {
                addr1=splithead($0);
                split(addr1,addr2," ");
                addr=addr2[2]" "addr2[3]" "addr2[4]" "
                addr2[5]","addr2[1]" ' "addr2[6];
                return addr;
               }
```

5.5.1 抽出処理対象範囲の選択

5.4.1項と同じように「addr1=splithead($0);」でDate項の抽出処理対象範囲を選択する。

例

Date: Mon, 10 May 1999 10:10:10 +0900

この範囲を「addr1」の値とする

5.5.2 スペースの照合とaddr1の分割

文字列操作関数「split(addr1,addr2," ")」は、5.5.1項で選択された文字列「addr1」を区切り文字に指定した「" "」（スペース）に従って文字列「addr2」に分割する。

結果、addr2[1]=Mon,、addr2[2]=10、addr2[3]=May、addr2[4]=1999、addr2[5]=10:10:10、addr2[6]=+0900に分割される。

5.5.3 分割した文字列の並べ替え

「addr=addr2[2]" "addr2[3]" "addr2[4]" "addr2[5]", "addr2[1]" ' "addr2[6];」と記述することで、前節で分割した各文字列を「, 」カンマで区切り「日 月 年 時間」,「曜日」,「時差」の順に配列しなおし文字列「addr」の値とする。

例　10 May 1999 10:10:10, Mon, '+900

5.5.4 定義関数の戻値

定義関数の戻値を返す「return addr」によって定義関数「day」の値は抽出された日付となり、5.2.3項の変数「date」の値は日付となる。

5.6 Subject項の照合と抽出

Subject項の中から標題となる文字列を抽出するために以下の関数「splithead」を定義する。この関数は、From項及びDate項の照合と抽出の中でも使われている。

定義関数「splithead」

```
function splithead(ptr) {
                m=index(ptr,": ")+2;
                addr1=substr(ptr,m);
                return addr1;
               }
```

5.6.1 抽出処理対象範囲の選択

文字列操作関数「m=index(ptr,":")+2」は文字列の中で「:」コロンが存在する位置に2を足して「:」コロンの後から二文字目の位置をmの値とする。このときスペースも一文字と数える。

例　subject:こんにちは

└──この位置をmの値とする

次に文字列操作関数「addr1=substr(ptr,m)」は文字列のmの位置から文字列の終端を示す改行コードまでの範囲を選択する。この範囲の文字列を「addr1」の値とする。

例　subject: こんにちは

この範囲を「addr1」の値とする

5.6.2 定義関数の戻値

定義関数の戻値を返す「return addr」によって定義関数「splithead」の値は抽出された標題となり、5.2.4項の変数「subject」の値は標題となる。

６　プログラムの実行

この章では、プログラムの実行方法を説明する。

６.１ AWKプログラムの準備とMS-DOSプロンプトの起動

コンピュータのハードディスク上にプログラミング言語と電子メール検索プログラムを準備する。今回は、Ｃドライブにawkというフォルダを作成し、プログラミング言語「JGAWK.EXE」と電子メール検索プログラム「mail.awk」を入れておく。JGAWK.EXEはMS-DOS環境下で稼動するので、MS-DOSプロンプトを起動する（図４参照）。次に、「C:¥awk>」の後に、キーボードから「JGAWK.EXE」と入力して「Enter」キーを押し、JGAWKを起動する（図５参照）。

６.２ 電子メール検索プログラム「mail.awk」の実行

AWKプログラムは２通りの実行方法がある。

例

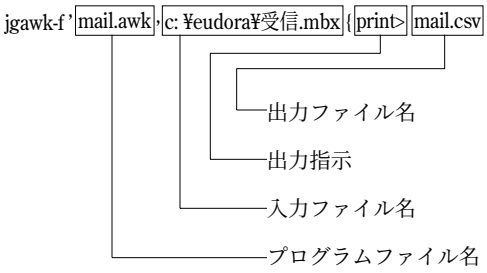
1. AWK’パターン-アクション’ 入力ファイル名

2. AWK－f’プログラムファイル名’ 入力ファイル名

１の方法は、指定したパターン－アクションを入力ファイルに対して実行する。２の方法は、ファイルにプログラムを蓄積し、このファイルを読込み実行する。

今回のプログラム「mail.awk」を実行する場合、２の方法で

と入力し、「Enter」キーを押す（図６参照）。入力ファイルに対しての処理が終了すると、



電子メール検索出力ファイル（mail.csv）上に処理結果を出力する。これによって、AWKプログラムの処理が終了する。

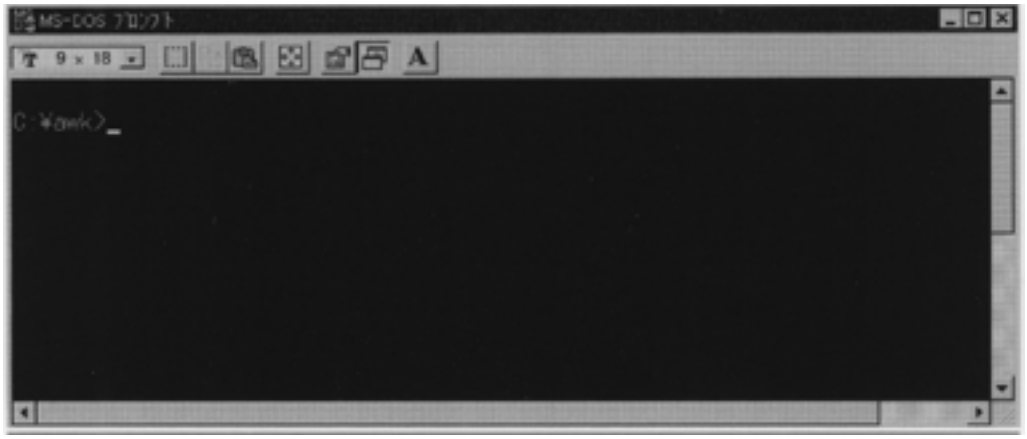


図４　MS-DOSプロンプトの起動

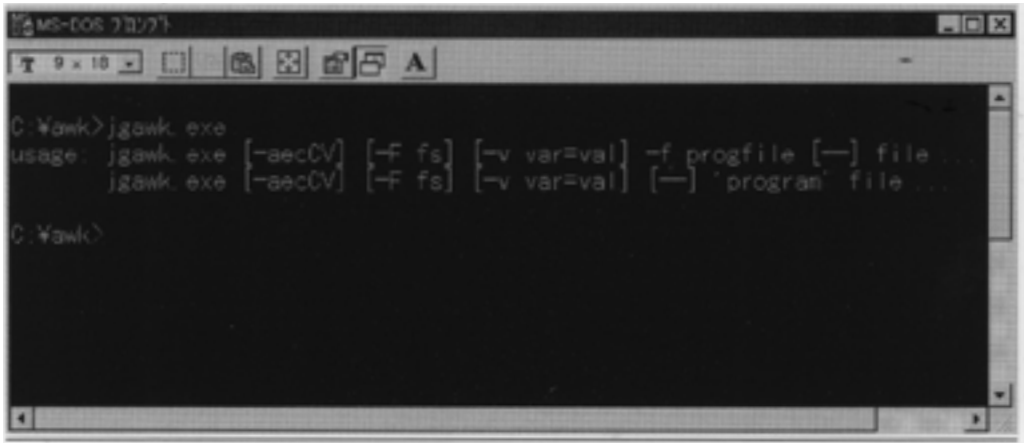


図５　プログラム言語「JGAWK.EXE」の起動

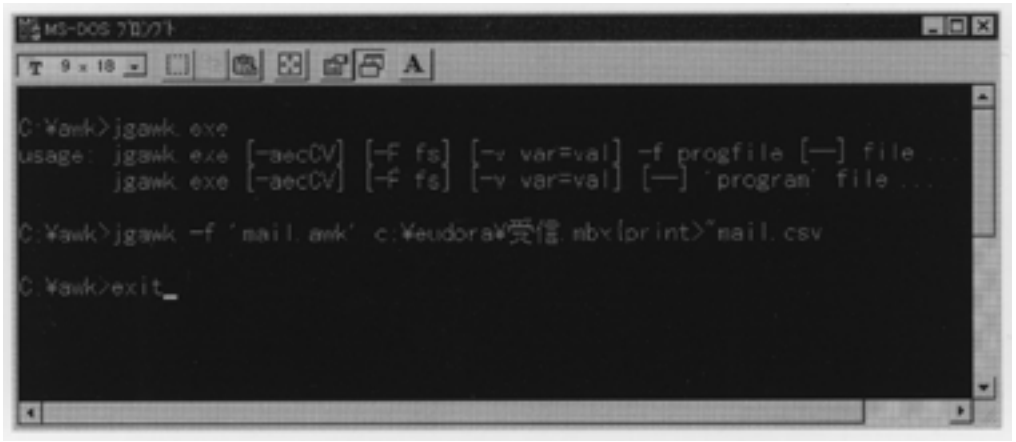


図６　電子メール自動集計プログラム「mail.awk」の実行

６.３　MS-DOS プロンプトの終了

「C:¥awk＞」の後に、キボードから「exit」と入力して「Enter」キーを押し、MS-DOSプロンプトを終了する(図７参照)。

６.４　MS-Excel の起動

電子メール検索プログラムの出力結果が電子メール検索出力ファイル（mail.csv）に対して集計処理を行うために、MS-Excel（以下Excelとする）を起動する。Excelへの入力データは、「CSV形式」で記述されているため、直接Excelの入力となり得る。

前節で出力した「mail.csv」ファイルのアイコンをダブルクリックすると（図８参照）

Excelが起動し、セル内にデータが表示（図９参照）される。

「CSV形式」ファイル

CSVル形式は、「Comma Separated Value format」の頭文字をとったものである。この

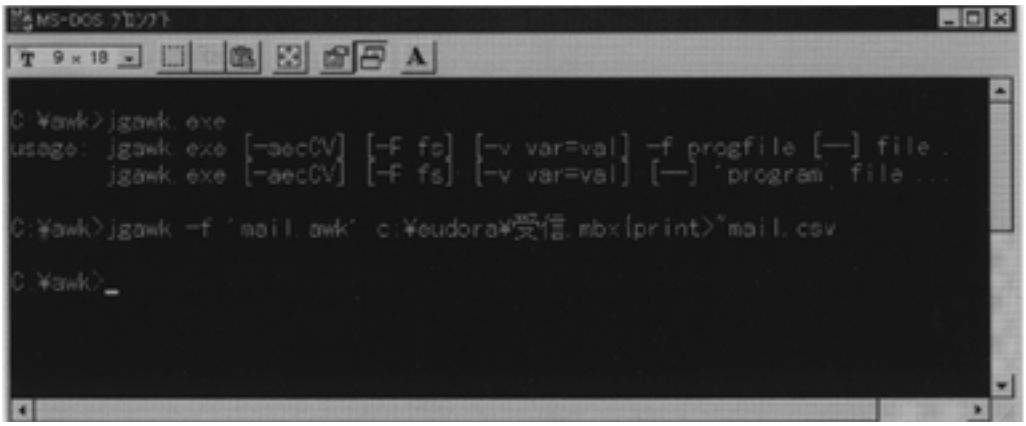


図７　MS-DOS プロンプトの終了

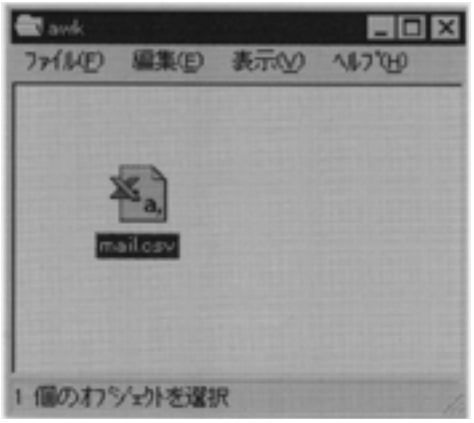
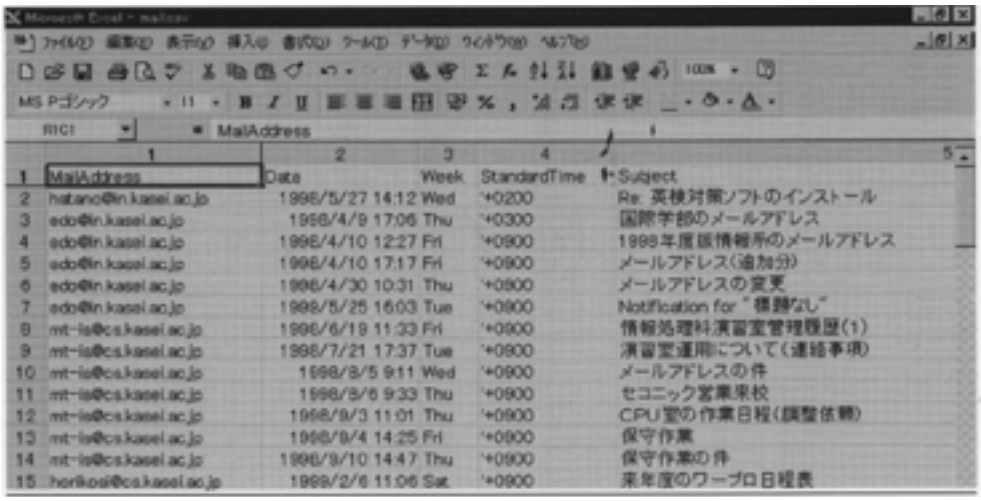


図８　ファイルのダブルクリックによる Excel の起動



	1	2	3	4	5
1	MailAddress	Date	Week	StandardTime	Subject
2	hatano@in.kasei.ac.jp	1998/5/27 14:12	Wed	+0200	Re: 英検対策ソフトのインストール
3	edo@in.kasei.ac.jp	1998/4/9 17:06	Thu	+0300	国際学部のメールアドレス
4	edo@in.kasei.ac.jp	1998/4/10 12:27	Fri	+0900	1998年度版情報所のメールアドレス
5	edo@in.kasei.ac.jp	1998/4/10 17:17	Fri	+0900	メールアドレス(追加分)
6	edo@in.kasei.ac.jp	1998/4/30 10:31	Thu	+0900	メールアドレスの変更
7	edo@in.kasei.ac.jp	1998/5/25 16:03	Tue	+0900	Notification for "標題なし"
8	mt-is@cs.kasei.ac.jp	1998/6/19 11:33	Fri	+0900	情報処理科演習室管理簿歴(1)
9	mt-is@cs.kasei.ac.jp	1998/7/21 17:37	Tue	+0900	演習室運用について(連絡事項)
10	mt-is@cs.kasei.ac.jp	1998/8/5 9:11	Wed	+0900	メールアドレスの件
11	mt-is@cs.kasei.ac.jp	1998/8/6 9:33	Thu	+0900	セコニック営業来校
12	mt-is@cs.kasei.ac.jp	1998/8/3 11:01	Thu	+0900	CPU室の作業日程(調整依頼)
13	mt-is@cs.kasei.ac.jp	1998/8/4 14:25	Fri	+0900	保守作業
14	mt-is@cs.kasei.ac.jp	1998/8/10 14:47	Thu	+0900	保守作業の件
15	hori-osi@cs.kasei.ac.jp	1999/2/6 11:06	Sat	+0900	来年度のワープロ日程表

図９　Excel での表示

ファイルは、列のデータが「,」（カンマ）で区切られ、行のデータは改行コードで区切られていている。これをExcelに変換すると、カンマで区切られた列のデータが各セルに入力される。

6.4 Excelの終了

Excelの表形式で表示されたデータを、電子メール自動集計処理出力ファイル（mail.xls）に保存して（図10参照）、実行を終了する（図11参照）。

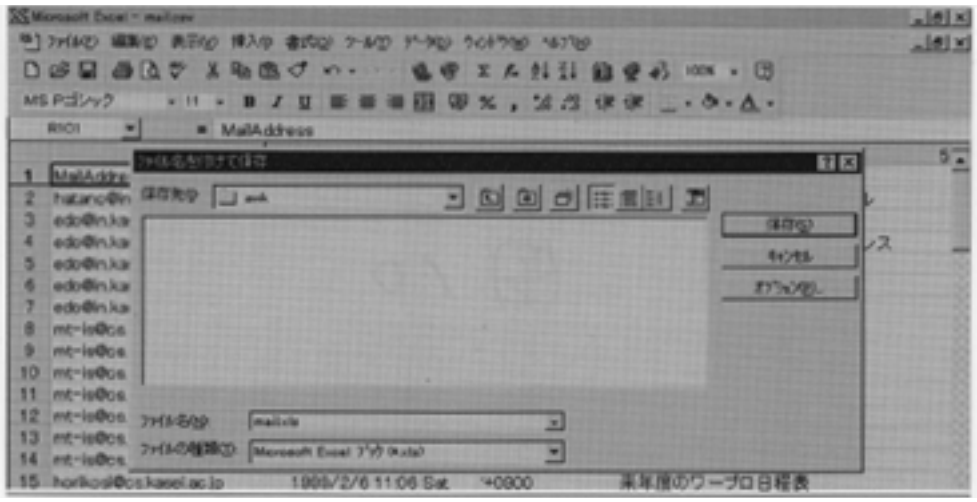


図10　Excel形式での保存

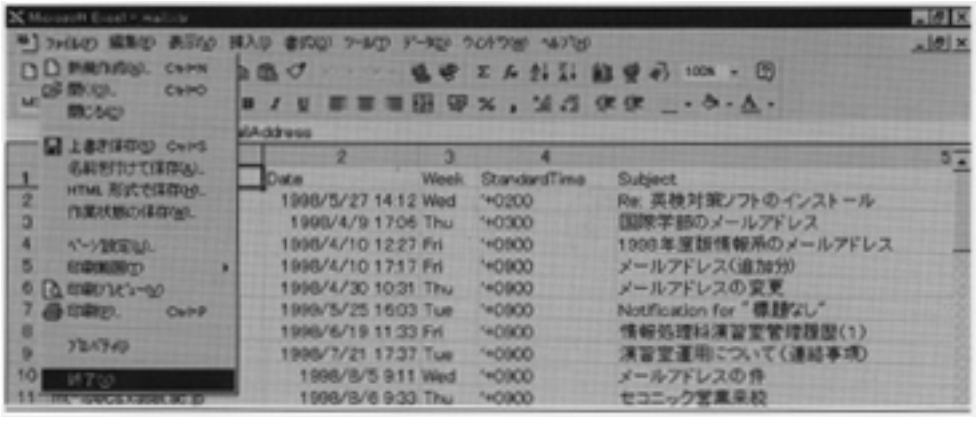


図11　Excelの終了

７　出力結果

７.１ 電子メール検索出力ファイル「mail. csv」の印字出力

例

MailAddress ,Date ,Week ,StandardTime ,Subject
hatano@in.kasei.ac.jp ,27 May 1998 14:12:09,Wed,'+0200 ,Re: 英検対策ソフトのインストール
edo@in.kasei.ac.jp ,09 Apr 1998 17:06:18,Thu,'+0300 ,国際学部のメールアドレス
edo@in.kasei.ac.jp ,10 Apr 1998 12:27:39,Fri,'+0900 ,1998年度版情報系のメールアドレス
edo@in.kasei.ac.jp ,10 Apr 1998 17:17:15,Fri,'+0900 ,メールアドレス（追加分）
edo@in.kasei.ac.jp ,30 Apr 1998 10:31:31,Thu,'+0900 ,メールアドレスの変更
edo@in.kasei.ac.jp ,25 May 1999 16:03:50,Tue,'+0900 ,Notification for "標題なし"
mt-is@cs.kasei.ac.jp ,19 Jun 1998 11:33:36,Fri,'+0900 ,情報処理科演習室管理履歴（1）
mt-is@cs.kasei.ac.jp ,21 Jul 1998 17:37:08,Tue,'+0900 ,演習室運用について（連絡事項）
mt-is@cs.kasei.ac.jp ,5 Aug 1998 09:11:00,Wed,'+0900 ,メールアドレスの件
mt-is@cs.kasei.ac.jp ,6 Aug 1998 09:33:33,Thu,'+0900 ,セコニック営業来校
mt-is@cs.kasei.ac.jp ,3 Sep 1998 11:01:20,Thu,'+0900 ,ＣＰＵ室の作業日程（調整依頼）
mt-is@cs.kasei.ac.jp ,4 Sep 1998 14:25:30,Fri,' +0900 ,保守作業
mt-is@cs.kasei.ac.jp ,10 Sep 1998 14:47:58,Thu,'+0900 ,保守作業の件

７.２ 電子メール自動集計処理出力ファイル「mail.xls」の印字出力

例

MailAddress	Date	Week	StandardTime	Subject
hatano@in.kasei.ac.jp	1998/5/27 14:12	Wed	'+0200	Re: 英検対策ソフトのインストール
edo@in.kasei.ac.jp	1998/4/9 17:06	Thu	'+0300	国際学部のメールアドレス
edo@in.kasei.ac.jp	1998/4/10 12:27	Fri	'+0900	1998年度版情報系のメールアドレス
edo@in.kasei.ac.jp	1998/4/10 17:17	Fri	'+0900	メールアドレス（追加分）
edo@in.kasei.ac.jp	1998/4/30 10:31	Thu	'+0900	メールアドレスの変更
edo@in.kasei.ac.jp	1999/5/25 16:03	Tue	'+0900	Notification for "標題なし"
mt-is@cs.kasei.ac.jp	1998/6/19 11:33	Fri	'+0900	情報処理科演習室管理履歴（１）
mt-is@cs.kasei.ac.jp	1998/7/21 17:37	Tue	'+0900	演習室運用について（連絡事項）
mt-is@cs.kasei.ac.jp	1998/8/5 9:11	Wed	'+0900	メールアドレスの件
mt-is@cs.kasei.ac.jp	1998/8/6 9:33	Thu	'+0900	セコニック営業来校
mt-is@cs.kasei.ac.jp	1998/9/3 11:01	Thu	'+0900	ＣＰＵ室の作業日程（調整依頼）
mt-is@cs.kasei.ac.jp	1998/9/4 14:25	Fri	'+0900	保守作業
mt-is@cs.kasei.ac.jp	1998/9/10 14:47	Thu	'+0900	保守作業の件

8 考　察

システムの作成にあたっては、厳密に規定された項目の判定とそこに含まれる文字列の抽出を、AWKを用いることにより、簡単なプログラムで処理することが出来た。さらに、この出力をExcelを利用することにより、プログラムを作成することなく表形式で出力することが出来た。

この結果は、膨大なメールボックスの中から、利用者が必要な情報のみを取り出す道具として役に立つと思われる。ただし今回のシステムは、メールボックス内のヘッダーに着目し、その中の特定の項目だけを抽出するにとどまっている。今後は、他の様々な情報についても、利用者の求める情報だけを簡単で、迅速に、見やすい形で出力できるシステムへと拡張していく予定である。

9 謝　辞

今回のプログラム作成にあたり、ご指導、ご協力を頂いた本学、国際学部比較文化学科海老澤成享教授と短期大学部情報処理科山野井一夫講師に感謝いたします。

参考文献

1)「RFC822: Standard for ARPA InternetText Messages」1982.5　David H. Crocker著

2)「sendmail解説」1994.8　Bryan Costales、Eric Allman、Neil ricket著　鈴木克彦訳　オーム社

3)「標準インターネット教科書」1996.7　安田浩監修　アスキー

4)「ａｗｋでプログラミング」1996.5　植村富士夫、富田浩之著　オーム社

5)「プログラミング言語ＡＷＫ」1997.7　Alfred V.Aho、Peter J.Weinberger、Brian W.Kernighan著　足立高徳訳　トッパン