GNU Emacsen における複数の MUA で共有可能な Internet Message のための汎用部品の開発

Development of Internet Message components for multiple MUAs on GNU Emacsen

守岡 知彦 [1)]
MORIOKA Tomohiko

1) 京都大学人文科学研究所附属漢字情報研究センター
（〒 606-8265 京都市左京区北白川東小倉町 47 番地　E-mail: tomo@kanji.zinbun.kyoto-u.ac.jp）

ABSTRACT. MIME introduces a structured Internet message format. It can represent various data formats and various coded character sets. MIME thus represents an important part of Internet message processing. Conventional message user agents (MUAs) which process Internet messages have different characteristics. The quality of the user interface depends on the user's preference or habits. In this regard, it may be preferable to allow the user to select a good combination of the MUA and a separate MIME module. For the purpose, we are developing a set of Emacs Lisp library mainly for MIME, named FLIM and SEMI.
This research and development is a continuation of the development of FLIM/SEMI. In the project, we made a set of renewal versions of FLIM/SEMI which have some good features of their variants, and implemented the new OpenPGP implementation and other features. In addition, we arranged and cleared the FLIM/SEMI API. To merge into GNU Emacs 21, we rewrote FLIM/SEMI and developed MIME-capable RMAIL.

1 背景

GNU Emacs/XEmacs は UNIX 系 OS において非常に普及した editor を中心とした統合環境であり、現在では UNIX 系 OS にとどまらず、Windows, Macintosh, OS/2, BeOS など多くの OS にも移植されている。GNU Emacs/XEmacs は編集操作や記号処理を得意とする Emacs Lisp 言語による非常に高度な拡張可能性を備えており、Emacs Lisp で記述された多くの application program が利用されている。Mail reader や news reader（以下、これらを総称して MUA (Message User Agent) と呼ぶことにする）も古くから作られてきており、現在もさまざまな実装が存在している。

各 MUA は各々の特色を持つが message 形式や MTA (Message Transfer Protocol) との間での protocol などは MUA 依存性が低いといえる。MIME (Multipurpose Internet Mail Extensions) の登場以降のさまざまな media-type や文字符号など message 形式の多様化や、さまざまな認証方式や protocol に対応する場合、個々の MUA で対処するよりも複数の MUA で共有可能な Internet Message のための汎用的な library を開発・保守する方が効率的であると考えられる。

このような考えに立って、我々は FLIM/SEMI と呼ぶ Internet Message のための汎用部品を開発してきた。FLIM/SEMI には標準的実装だけでなく同一仕様の置き換え可能なさまざまな変種が作られてきているが、このことは新しい protocol や利用者の嗜好に対応する上で有効な方法であると考えられる。FLIM/SEMI の枠組みを用いることにより MUA 開発者は少ない労力で高機能な MUA を作ることができ、利用者も多様な実装の組み合わせを選択可能である。このような FLIM/SEMI の枠組みの有用性は実際に Wanderlust や cmail, Semi-gnus 族などの FLIM/SEMI に基づく高機能な MUA が開発され普及していることによって実証されていると考えられる。

特に最近 Wanderlust が高機能な MUA として人気を博しており、FLIM/SEMI はこれらの MUA の動

作基盤としてより良い実装の供給が期待されている。初心者が最初に導入する場合や各種 distribution での便宜を鑑みれば、ある高機能性と安定性を兼ね備えた FLIM/SEMI の標準実装を提供することは重要である。また、最初から GNU Emacs に含まれるようになれば FLIM/SEMI 系 MUA を使うのに FLIM/SEMI を利用者が別途導入する必要はなくなるだけでなく、現在の GNU Emacs の貧弱な mail/news 環境を大幅に改善することができる。

2 目的

GNU Emacs/XEmacs における GNU Emacsen における複数の MUA (Message User Agent) で共有可能な Internet Message のための汎用部品の開発およびその標準化を行い、FLIM/SEMI を利用する各種 MUA の動作環境の改善を図るとともに、各種 OS, distribution 等での導入時の利便性を図る。

また、GNU Emacs への統合を行う上で必要となる作業を行い、GNU Emacs の標準環境における mail/news 機能を改善する。

3 開発の概要

本研究開発では既存の FLIM/SEMI の標準実装および変種を整理統合し、FLIM/SEMI の標準実装の近代化をはかるとともに、現在開発中の GNU Emacs 21 への対応および統合作業を行った。

(1) FLIM

本プロジェクトで開発した FLIM API およびその実装は図 1 に示すようなモジュールからなっている。各モジュールの機能は以下のようなものである：

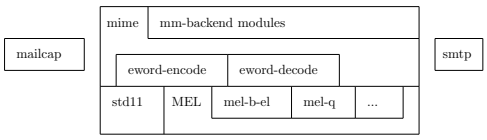

図 1: FLIM の構成

std11 MAIL(STD 11) [2] 形式に基づく解析処理などを行う

MEL 転送用符号化法 (Content-Transfer-Encoding) の符号化・復号化処理を行う

eword-encode 見出しの符号化処理を行う

eword-decode 見出しの復号化処理を行う

mime mime-entity に関する機能を提供する（mime-entity モデルを実現するための核）

mm-backend モジュール群 mime-entity に関する処理を実現する (mime-entity モデルにおける mm-backend)

smtp SMTP (Simple Mail Transfer Protocol) [5] によるメッセージの送信処理を行う

mailcap mailcap[1] ファイルの解析処理を行う

FLIM API が想定する実装モデルでは、FLIM (FLIM) は **FLIM kernel** と **mm-backend** 群で構成される。

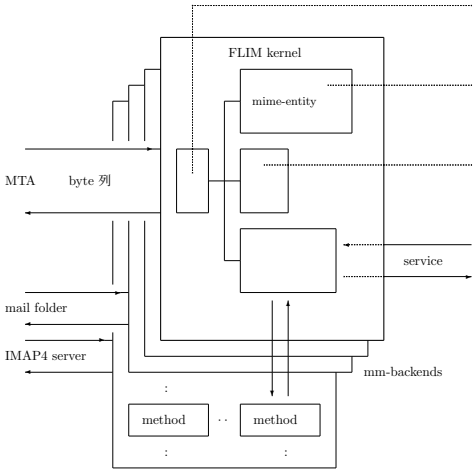

図 2: FLIM API の実装モデル

FLIM の標準実装では、FLIM kernel および mm-backend 群は CLOS に似た文法を持つオブジェクト指向言語機能を提供するモジュールである **luna** で記述される。luna を設けたことにより、オブジェクトの記述が簡潔になり、また、mm-backend においてより高度な拡張が可能になった。

a) パラメータ拡張

MIME 形式に従って、メッセージにファイルを添付することを考えてみよう。この場合、MIME では Content-Disposition 欄 [7] の filename パラメータを用いてファイル名を表現することができる。しかしながら、パラメータ *parameter* の値 *value* は encoded-word を含むことはできない。このため、このままでは表現可能なファイル名は US-ASCII で表現可能な文字列に限られることになる。これを解決するには符号化結果が MIME 関連の欄における字句要素である “token” になるような符号化法を作れば良いが、符号

化すると従来よりも長い *parameter value* を扱う必要が出て来る。また、*parameter* 一般の用途としては、言語を指定したい場合もあるかも知れない。例えば、音声出力を目的としている場合などが考えられる。このような要求に答えるために、*parameter value* に関する拡張仕様が提案された [3]。これをここでは『拡張パラメータ』(parameter extension) と呼ぶことにする。拡張パラメータの機能には以下のものがある：

1) 文字符号 (MIME-charset) の指定

2) 言語の指定

3) 長いパラメータの分割表現

拡張パラメータを定めた最新の RFC は現在のところ RFC 2231[3] である。この RFC では拡張パラメータだけでなく、encoded-word における言語の指定に関する拡張も定めている。

さて、このパラメータ拡張は、仕様の制定が遅れたこともあって FLIM でも対応が遅れ、鈴木圭一氏の手による RFC 2231 に対応した FLIM 実装は存在するものの、FLIM 標準実装への統合は未だ行なわれていない。

この鈴木圭一氏によるパラメータ拡張に対応した既存の FLIM 実装を検討した結果、不必要に複雑化していると判断し、本プロジェクトでは新たな実装を開発することにし、小林修平氏がその作業を行った。

小林修平氏の手による新実装は現在テスト中であり、問題点を洗い出し、修正した上で、FLIM 標準実装に統合する予定である。

b) HMAC: Keyed-Hashing for Message Authentication (RFC2104)

Internet の mail 関連の protocol（SMTP, POP3, IMAP4 等）の認証方式として、共通の枠組である SASL が使われるようになった。このため、Internet Message を扱う library である FLIM としても SASL に対応する必要が生じた。このため、本プロジェクトでは SASL 関連の API と実装を開発することにし、上野乃毅氏らがその作業を行った。

SASL に基づく認証方式では MD5 や SHA-1 などの cryptographic hash 関数を応用した HMAC, 特に HMAC-MD5 という方法が広く使われているので、本プロジェクトではまずこれに関する部分を作成することにした。

小林修平氏の調査により、既に Emacs 上で HMAC-MD5 を計算する rfc2104.el が存在することが判明したが、仕様上の問題があったため本プロジェクトでは独自の実装を開発することにし、小林修平氏らがその作業を行った。その結果、本プロジェクトでは hmac-def.el, hmac-md5.el, hmac-sha1.el, hex-util.el を開発した。[1]

また、MD5 や SHA-1 を Emacs 上で計算する md5.el や sha1.el も既に存在したが、Copyright 上の問題（C 言語による実装の “direct translation” であると書かれている）などから、本プロジェクトでは仕様書だけを基に独自に作成することにし、小林修平氏らがその作業を行った。

本プロジェクトにおいて小林修平氏らは MD5 や SHA-1 の実装として Emacs-Lisp 版と Dynamic-Loading module 版を作成した。しかし、Emacs の Dynamic-Loading 拡張は将来性が不透明なことから Emacs への統合作業は行なわなかった。また、GNU Emacs 21 の pretest 版を入手した段階で MD5 が新たに Emacs の built-in 関数となったことが判明したため、最終的には SHA-1 の Emacs-Lisp 版 (sha1.el) のみ統合作業を行なった。

(2) SEMI

SEMI は FLIM の機能を利用して、利用者に Internet message の構造を扱うための対話的な user interface を提供するための層で、message の表示や entity の再生処理などを行う MIME-View と message の編集を行うための MIME-Edit からなる。

SEMI は MIME 処理を行うための user interface のための汎用的な核として、さまざまな MUA (Message User Interface) で利用することを考慮している。MIME-View の表示および再生処理は、『実行状況モデル』に基づき、

- entity の構造に関する情報
- MUA の種類などの実行環境に関する情報
- 操作の種類
- SEMI 標準のものおよび利用者が mailcap [1] 等で設定した処理条件

といった制約を満たした実行状況に関する情報を捜し出し、それに応じて行われる。このような方法により、message の構造を実行環境と利用者の意図を反映した形で解釈することを可能にしている。再生処理に関しては例示インターフェース的な考え方による、利用者の操作に対する適応も行っている。表示に関してもこの考え方を採り入れる予定である。

図 3 に SEMI API が想定する実装モデルを示す。この図に示されるように、SEMI API はメッセージ解釈部 (SEMI) を **SEMI kernel** と、**presentation-method** / **acting-method** という拡張可能なモジュールからなるものと想定している。

今回の SEMI に対する開発として大きなものは、OpenPGP 関連の機能の再実装である。従来の SEMI でも PGP/MIME を扱うことを主たる目的として

[1] 小林注：rfc2104.el は最終的な結果を hexadecimal form で返すが、RFC 2104 的には binary form を返すのが正しい。HMAC-MD5 を応用した SASL の一方式 CRAM-MD5 では hexadecimal form を使用するために、rfc2104.el の間違いは表面化していない（小林修平氏は以前に rfc2104.el の作者にこのことを指摘した）。

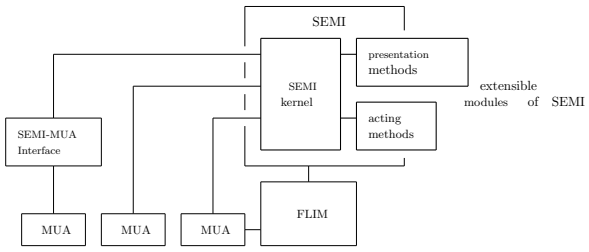

図 3: SEMI API の実装モデル

PGP Version 2 用の実装があったが、OpenPGP が規定され GnuPG などの実装が普及している現状に十分に追随していなかったといえる。今回、上野乃毅氏が中心となって “PGG” と呼ぶ電子署名・暗号を扱うためのモジュール群を開発した。これは FLIM と同様に luna を使って記述されており、backend による拡張が可能な設計になっている。

(3) GNU Emacs 21 への対応および統合作業

Free Software Foundation がリリースしている GNU Emacs に附属する MUA は全て従来 MIME に対応していなかった。このため、数年前、著者に対して Free Software Foundation から tm[9] を GNU Emacs に統合する作業の要請があり、著者らはこれに対する作業を開始した。しかし、まず技術的な作業を行う前に法的な権利関係の整理作業を行う必要があり、これが概ね終了したのは比較的最近のことである。

２年程前より著者は技術的な作業を開始したが、いくつかの難問から作業はしばらく停止していた。それは RMAIL の MIME 化の問題と FLIM/SEMI で用いている共通ライブラリ “APEL” の問題である。

a) RMAIL の MIME 化

RMAIL は GNU Emacs における標準の mail reader であり、GNU Emacs 21 と FLIM/SEMI の統合において RMAIL を MIME 化することは非常に重要である。しかしながら RMAIL を MIME 化することは技術的に困難であることが判明した。

MIME ではメッセージは entity を単位とする木構造のデータ表現になっている。画像などの情報は符号化されて格納されており、それを復号しなければ表示できない。また、音声のようなそもそも画面表示できないような情報も含まれる。テキスト情報だけで構成される場合でも、entity 毎に異なる文字符号を使うことが可能であり、メッセージ全体をプレーン・テキストと見做して符号変換することはできない。このため、MIME メッセージを扱うためにはメッセージのネットワーク表現とは別にメッセージを表示・操作するためのバッファを設ける必要がある。

しかしながら、RMAIL はメッセージ・フォルダー全体が１つのファイルになっており、このファイル全体をバッファに格納し、表示すべきメッセージの入った領域を選択して表示するような仕組みになっている。即ち、メッセージのネットワーク表現を格納したバッファをそのまま表示する訳である。また、RMAIL における操作はこのバッファの中で行うのが基本であり、RMAIL が管理するバッファとは別にメッセージを表示・操作するためのバッファを設けるのは簡単ではない。

さらに悪いことに、RMAIL におけるさまざまな操作を実現する各関数はばらばらに RMAIL のバッファを操作するような実装になっており、RMAIL の挙動を変えるためには非常に多くの部分を変更する必要がある。

しかし、RMAIL の MIME 化が GNU Emacs 21 に取り込まれるためには、大規模な修正は望ましくなく、修正意図の理解が容易な小規模な修正である必要がある。

このような相矛盾する要求を満たすべく、RMAIL の MIME 化作業を行った。

ここで行った作業は主にメッセージのネットワーク表現を格納したバッファと表示・操作用バッファを分離可能にするためのものである。RMAIL においてメッセージ・フォルダーを格納したバッファはバッファローカル変数 ‘rmail-buffer’ に格納され管理されて来たが、本プロジェクトではこの変数で指されるバッファをメッセージのネットワーク表現を格納したバッファと見做すことにし、これとは別に表示・操作用バッファを指すバッファローカル変数 ‘rmail-view-buffer’ を設けた。そして、RMAIL の中で従来 ‘rmail-buffer’ を参照していた部分をその意図に応じて、‘rmail-buffer’ と ‘rmail-view-buffer’ に使い分ける変更を行った。

また、MIME モードでも従来モードの双方で RMAIL が動作するようにした。従来モードでは ‘rmail-buffer’ と ‘rmail-view-buffer’ は同じバッファを指し、従来と同様の挙動になる。MIME モードでは ‘rmail-buffer’ とは別に ‘rmail-view-buffer’ が作られる。

b) APEL の除去

GNU Emacs およびその変種の Emacs Lisp[4] の仕様および機能は基本的な Lisp の言語仕様に関しては概ね共通している。しかしながら、GNU Emacs [6] と XEmacs [8] の間では細かな字句・構文の差があり、また、menu, event, 文字、face, font, 領域の属性、文字列の属性など多岐に渡って大きな構文・機能の差異がある。XEmacs では画像を扱うことが可能であり、XEmacs で拡張された機能も少なくない。個々の関数の有無や界面の拡張などは版によっても異なる。このようなことから、多くの環境で動作する実装を作成することはそれほど自明ではない。

地域化・多言語化・国際化に関しても実装間で大きな差異がある。Emacs 19.34 や XEmacs 19.15 では文字は 0 から 255 までの数字で表現され、フォントの設定によって ISO 8859-1 (Latin-1) 等の 8bit 1byte 文字符号を扱えるようになっていた。MULE は GNU

Emacs を拡張することにより、さまざまな図形文字集合を同時に扱えるようにした。この MULE 機能は Emacs 20.1 で取り込まれた。XEmacs 20 以降では MULE 機能を使うか使わないかが XEmacs のコンパイル時に決定できるようになった。このように現在 MULE 機能を持つ emacs 実装としては本来の MULE, Emacs 20, XEmacs-MULE がある。これらの機能は本質的に同様のものであるが、MULE 機能に関する API (Application Program Interface) は互いに異なっている。Emacs 20 と XEmacs 20 with MULE は比較的似ているが、XEmacs-MULE では文字が 1 級オブジェクト (first class object) であるのに対し、Emacs 20 では文字は整数で表現されるなどの差がある。Emacs 20 でも Emacs 20.2 までと Emacs 20.3 以降では大きな差がある。

このため、FLIM/SEMI では APEL[10] と呼ぶシステムを作成し、これを用いることによって可搬性を実現してきた。APEL は

- 最新の emacs 実装で追加された関数をそれ以前の emacs 実装上でエミュレートする
- MULE 機能に関する抽象を各種 emacs 実装上で利用可能にする
- 可搬性のある共通ライブラリを提供する

といった機能を提供するものである。

APEL の機能を用いることで、APEL を使わない場合に比べると、可搬性の高いプログラムを書くのが容易になる。Emacs 実装間の差異に対する対処を個々のプログラム毎に行うよりも APEL に集約する方が、各プログラムの実現や保守の点で優れている。また、利用者にとっても個々のプログラム毎の Emacs 実装間の差異に対する対処部分が APEL に共通化されることで、記憶資源の節約につながる。このような利点から、現在、APEL は FLIM / SEMI API の実装や関連システムに留まらず、日本語入力システムの **SKK 10** や IRC client の **Liece** などでも使われるようになった。

しかしながら、最新の emacs 実装を対象にする場合、APEL は本来不要であるはずである。また、APEL の機能の内、エミュレーションに関する部分はコンパイル時に環境を検査し、標準の関数を置き換えたり足りない関数を定義するようなマクロを多用しているが、これが emacs 実装の構築時に問題を起こす可能性がある。このため、APEL の機能の内エミュレーションに関する部分を FLIM/SEMI の標準実装から除去する作業を行った。

c) GNU Emacs 21 との統合作業の現状

GNU Emacs 21 を対象にした RMAIL の MIME 化作業は概ね完了したがまだいくつかの問題点があり、GNU Emacs 21.1 に対する統合は完了しなかった。しかしながら、RMAIL に対する必要な修正は取り込まれることになりそうである。このため、GNU Emacs 21.1 に対して付加パッケージを作りそれを用いることにより、MIME 化した RMAIL が利用可能になる。

また、問題点の修正は今後も継続し、引続き GNU Emacs 21.2 との統合を目指すつもりである。

(4) XEmacs におけるパッケージ化作業

XEmacs 19.13 以降の XEmacs には SEMI の前身にあたる tm が附属していた。XEmacs 21.1 以降では各種アプリケーションはパッケージに分離されたが、tm もその 1 つとして提供されて来た。しかしながら、既に tm は obsolete なものであり、現在の要求を十分に満たさない。

このため、従来の tm の利用者に配慮した形で、tm を SEMI に置き換える作業を開始した。

(5) Meadow

Meadow は宮下尚氏が開発している Windows 95/98/NT/2000 で動作する GNU Emacs で、多言語表示が可能であり、また、いくつかの Windows 特有の拡張を行っている。

Meadow の過去の版には初期の SEMI が附属していたが、今回、本プロジェクトでは最新の安定版の FLIM/SEMI を統合した Meadow の開発を宮下尚氏に依頼し、それを収録したインストール用 CD-ROM を開発した。

この CD-ROM はハードディスクにインストールすることなく Meadow と共に FLIM/SEMI と Wanderlust, T-gnus を利用可能であり、また、メニューによって簡単にハードディスクにインストールすることも可能である。

4 むすび

今回、本プロジェクトは FLIM/SEMI の各種変種の機能を統合し、認証機能の強化や OpenPGP 実装への対応などの以前に計画されていた機能を実現した新たな実装を開発した。また、API の整理も行った。

GNU Emacs 21 との統合を目指し、coding style を GNU Emacs の基準に適合させる作業や、APEL のエミュレーション関連の機能の除去を FLIM/SEMI に対して行った。また、RMAIL の MIME 化のための RMAIL の修正を行い、RMAIL において FLIM/SEMI の提供する高度な MIME 機能を RMAIL と調和する形で利用できるようになった。

また、XEmacs や Meadow などでのパッケージング作業も開始した。

今後の課題としては、引続き GNU Emacs 21 との統合作業を行うと共に、FLIM の諸機能のオブジェクト指向に基づく再設計や、SEMI の MIME メッセージ編集用ユーザーインターフェースの再実装などが挙げられる。また、Emacs Lisp 以外での FLIM/SEMI API の実現も興味深い課題である。

5 参加企業及び機関

(1) 参加機関

本研究開発は、主に京都大学人文科学研究所附属漢字情報研究センター（守岡知彦）、小林修平氏、早稲田大学大学院理工学研究科（上野乃毅氏）、京都大学情報学研究科（岡田健一氏）によって行われた。この他、FLIM/SEMI およびその前身の tm の開発に対して、AKITO Ishihara 氏、AMAKAWA Shuhei 氏、ARISAWA Akihiro 氏、Alastair Burt 氏、Alexandre Oliva 氏、Andy Piper 氏、Artur Pioro 氏、Dan Rich 氏、HANDA Kenichi 氏、HAYASHI Yoshiki 氏、Hunter Kelly 氏、Jari Aalto 氏、Jens Lautenbacher 氏、Joe Nuspl 氏、John S. Cooper 氏、KAWAGISHI Taro 氏、KON-NO Yoichi 氏、KOSEKI Yoshinori 氏、Kevin Broadey 氏、MAEDA Shugo 氏、MINOURA Makoto 氏、MIYASHITA Hisashi 氏、MURATA Hiroya 氏、MURATA Masahiro 氏、Marc Girod 氏、Martin Buchholz 氏、Mito 氏、NAKAJIMA Mikio 氏、Nakagawa Makoto 氏、Neal Becker 氏、OHTA Kazuhiro 氏、OKABE Yasuo 氏、OKAZAKI Tetsurou 氏、OKUNISHI Fujikazu 氏、Oscar Figueiredo 氏、Pekka Marjola 氏、Rob Kooper 氏、SANETO Takanori 氏、SHIONO Jun'ichi 氏、SUZUKI Keiichi 氏、Simon Josefsson 氏、Steinar Bang 氏、Steven L. Baur 氏、TAKAHASHI Kaoru 氏、TANAKA Akira 氏、TANAKA Shin'ichiro 氏、TERANISHI Yuuichi 氏、TSUCHIYA Masatoshi 氏、TSUKAMOTO Tetsuo 氏、UEHARA Shozo 氏、UENO Hiroshi 氏、William Perry 氏、YAMAGAMI Yoshiyuki 氏、YAMAOKA Katsumi 氏、YAMASHITA Junji 氏らが貢献している。

また、本研究開発には、京都大学人文科学研究所附属漢字情報研究センターのほか、通商産業省工業技術院電子技術総合研究所（当時；現 独立行政法人 産業総合技術研究所）の "m17n project", エイアンドエー株式会社（守岡家）の協力を得て、これらにネットワーク上および物理的な開発拠点を置いている。

6 参考文献

[1] Nathaniel S. Borenstein. A user agent configuration mechanism for multimedia mail format information. RFC 1524, The Internet Society, September 1993. Informational.

[2] David H. Crocker. Standard for the format of ARPA Internet text messages. RFC 822, The Internet Society, August 1982. STD 11 (Obsoletes RFC 733).

[3] N. Freed and K. Moore. MIME parameter value and encoded word extensions: Character sets, languages, and continuations. RFC 2231, The Internet Society, November 1997. Proposed Standard.

[4] Bil Lewis, Dan LaLiberte, and Richard Stallman. GNU Emacs Lisp Reference Manual. Free Software Foundation, 2.5 edition, May. for Emacs Version 20.3.

[5] Jonathan B. Postel. Simple Mail Transfer Protocol. RFC 821, The Internet Society, August 1982. STD 10.

[6] Richard M. Stallman et al. GNU Emacs version 21.1. ftp://ftp.gnu.org/gnu/emacs-21.1.tar.gz, October 2001.

[7] Rens Troost, Steve Dorner, and Keith Moore. Communicating presentation information in Internet messages: The Content-Disposition header field. RFC 2183, The Internet Society, August 1997. Proposed Standard (Updates RFC 1806).

[8] XEmacs. ftp://ftp.xemacs.org/pub/xemacs/.

[9] 守岡 知彦, 小林 修平, et al. tm (tools for MIME) version 7.106. ftp://ftp.jaist.ac.jp/pub/GNU/elisp/mime/tm-7.106.tar.gz, May 1997.

[10] 守岡 知彦, 田中 哲, 小林 修平, 山岡 克美, 中島 幹夫, et al. APEL (A Portable Emacs Library). ftp://ftp.jpl.org/pub/elisp/apel/.