

## Emacs による推敲ユーザインターフェース\*

3L-8

島谷安俊, 鈴木美紀, 五十嵐春雄†  
 沖電気工業株式会社‡

## 1 はじめに

最近、ワープロ技術の発達により高度な文書処理がコンピュータ上で簡単に行なえるようになってきた。企業内のドキュメントもほとんどワープロを使用したものになってきている。このような状況が進むにつれ、普段ワープロを使用しない人もワープロを使用してドキュメントを作成することになる。この場合、タイプミスやカナ漢字変換ミスといったことが起こりやすく、推敲機能をもったワープロの研究開発が行なわれている。しかし、このような推敲機能はシステムに組み込まれている場合が多い。

そこで、我々はまず UNIX<sup>1</sup> 上で、推敲文書としてテキストファイルを対象とした推敲機能を行なう推敲サーバを構築し、そのクライアントとして豊富な編集機能を持つ Emacs を使用したシステムを試作した。

推敲システムは、文書編集機能と一体になってはじめて使い勝手が良くなるといえ、一般に、推敲システムは文書処理システムに組み込まれている場合が多い。つまり文書処理の1つの機能として推敲機能が存在することになる。この場合、その推敲機能はその文書処理システムだけでしか使えず、また高度な推敲を行なおうとすると辞書が大きくなり1つのシステムに組み込むことが難しくなる場合がある。

そこで我々は、まず、UNIX 上で推敲サーバの構築を行なった。推敲機能を行なう部分をサーバ化することによりネットワーク内の CPU パワーの大きいマシン、あるいは負荷の少ないマシン上で推敲サーバを起動しておけば、各クライアントの負荷が少なくなり、推敲サーバの維持も軽減される。

次に、この推敲サーバを使う側である推敲クライアントを構築することになるが、前にも述べたように、編集機能と一体とならなければ使いにくいものになってしまう。

文書編集機能を作りそれに推敲機能を組み込むことは大変な作業となるため、この編集機能部を、Emacs を使用することにより作成の手間を省くことにした。Emacs を使用することにより、現在ではテキストファイルのみが推敲対象文書となる。

Emacs の特徴として、

- 豊富な編集機能
- Emacs Lisp による容易なカスタマイズ
- 任意のキーバインディング

などがある。

我々は、この Emacs Lisp による容易なカスタマイズに注目し、Emacs を作り直すことなく推敲機能を組み込むことを目指した。

## 2 Emacs によるユーザインターフェース

Emacs は、Emacs Lisp により、「バッファの制御」、「ウィンドウの制御」、「カーソルの制御」、「プロセス間通信」が可能である。

## 2.1 状態

本推敲インターフェース (suikou.el) は、図 1 に示す状態を持つ。

- 通信準備状態：推敲サーバへ接続する。
- モード選択状態：会話型かバッチ型かを選択する。
- 機能選択状態：推敲機能を選択する。さらに自動修正可能なものは、自動修正を行なうかどうかを指定する。
- メニュー選択状態：【モード選択】【機能選択】【推敲処理】【終了】を選択する。
- 推敲処理状態：推敲サーバへ推敲依頼し結果を表示する。
- 終了状態：終了する。

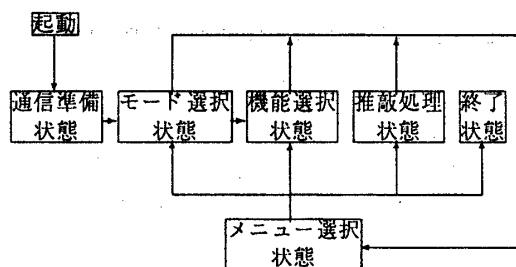


図 1: 状態遷移図

## 2.2 起動

Emacs 上で、『M-x suikou RET』を入力することにより、suikou.el がロードされ実行され、まず通信準備状態になり、推敲サーバへ接続する。接続が完了すると、モード選択状態へ入る。

\*User Interface of Proofreading with Emacs

†Yasutishi Shimatani, Miki Suzuki, Haruo Igarashi

‡Oki Electric Industry Co., Ltd.

<sup>1</sup>UNIX は AT&T の登録商標です。

### 2.3 モード選択状態

モード選択状態に入ると、モード選択用ウィンドウが表示され、「対話型」か「バッチ型」の選択をユーザに要求する。

### 2.4 機能選択状態

機能選択状態に入ると、機能選択用ウィンドウが表示され、推敲機能の選択をユーザに要求する。

### 2.5 メニュー選択状態

メニュー選択状態に入ると、メニュー選択用ウィンドウが表示され、状態の選択をユーザに要求する。この状態は、トップレベルの状態であり起動時以外では、終了以外の各状態の処理が終ると、この状態に戻る。

起動時だけは、「通信準備処理」、「モード選択状態」、「機能選択状態」の順に状態ががかわっていく。

### 2.6 推敲処理状態

推敲サーバから推敲結果を受信すると、以下の3つのバッファを作成する。

- ・結果バッファ 推敲結果文
- ・指摘情報バッファ 自動修正以外の指摘情報
- ・自動修正履歴バッファ 自動修正の指摘情報

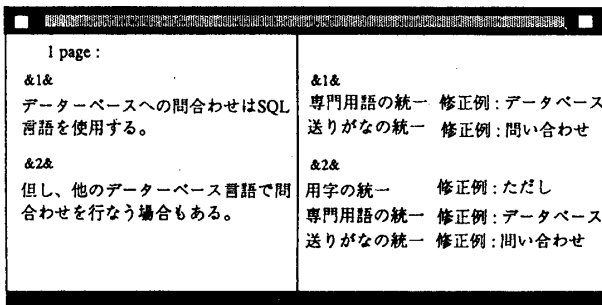


図2: 推敲画面

次に、図2のようにウィンドウを2つに分割する。左側のウィンドウは結果ウィンドウと呼び結果バッファを、右側のウィンドウは指摘情報ウィンドウと呼び指摘情報バッファを表示する。

図中「& 数字 &」は文番号を表し、自動的に結果および指摘情報バッファに挿入される。指摘情報がどの文に対するものかを分かりやすくするために、結果バッファの文番号の位置と、指摘情報バッファの文番号の位置が合うように表示される。

結果あるいは指摘情報ウィンドウで、サーチやポインタ移動コマンドにより画面のスクロールが起こった場合でも、結果バッファと指摘情報ウィンドウの文番号の位置が合うように、Emacsのサーチおよびポインタ移動コマンドが拡張されている。結果ウィンドウで、文字の挿入や削除により文番号の位置が変わった時は、画面のずれを直すコマンド（「C-1」）により、ずれを直すことができる。

しかし、文番号を表す「& 数字 &」が原文中にあったり、文番号を編集してしまうと誤動作を起こしてしまう。

suikou.el で追加したコマンドとして、入れ換えコマンドがある。この入れ換えコマンドは「C-i」にバインドされており、指摘情報ウィンドウ上で指摘項目にポインタを移動しこのコマンドを実行すると、左の結果ウィンドウに表示される結果バッファの内容が指摘情報に従って変更される。

例えば、

結果ウィンドウ	指摘結果ウィンドウ
&1& 但し、他の…	&1& 用字の統一 修正例: ただし

図3: 入れ換え前

と表示されている時、ポインタを指摘情報ウィンドウの指摘項目に移動し、「C-i」をタイプすると、

結果ウィンドウ	指摘結果ウィンドウ
&1& ただし、他の…	&1& 用字の統一 修正後: 但し

図4: 入れ換え後

となる。もう一度、「C-i」をタイプすると図3に戻る。

### 2.7 終了状態

このモードに入ると、結果バッファ、結果情報バッファ、自動修正履歴バッファがファイルに書き込まれ、使用したバッファを解放し終了する。

## 3 おわりに

UNIX 上での推敲サーバ/クライアントシステムにおいて、Emacsを使用した推敲クライアントの試作を行なった。Emacsを使用することにより、新たに編集機能を作成することなくEmacsの持つ豊富な編集機能を保ったまま推敲クライアントを作成することができた。また、Emacsに慣れている人にとっては、新たに編集コマンドを覚える必要がなくなった。

しかし、推敲文書がテキストファイルに限定されるため、現在のところあまり利用範囲は広くない。

今後は、TeX や roff といった UNIX 上での文書処理やパソコンなどのワープロからのフォーマット変換フィルターの検討を行なっていきたい。

## 参考文献

- [1] 「GNU Emacs Lisp Reference Manual」
- [2] Richard Stallman 著/竹内郁雄・天海良治監訳 「GNU Emacs マニュアル」 共立出版株式会社 (1988年)