cannaによる日本語入力

2004年3月29日

はじめに

日本語の文章を書く場合に、「日本語入力システム」が自分の意図した通りの変換をしてくれないと、そのことに気がとられて文章の作成に集中することが出来ないので、作業効率が大幅に落ちる。このことは、**慣れ**の問題も大きく、ある人には非常に使い易いシステムも、別なシステムに慣れている人には使いづらいことも多い。また、最近ではほとんどの「かな-漢字変換システム」が、学習機構を備えているので、使い込めば変換効率は上がる。更に、「日本語入力システム」は、自分の好みで**カスタマイズ**することが可能である。「UNIXでの日本語入力は使いづらい」という声を良く聞くが、これは、他のシステムに**慣れ**た人が、全く**カスタマイズ**せずに使った場合には、もっともな感想だと思う。

筆者は、ATOK3、EG-Brigde、ことえり、SKKなどの日本語入力を使ってきて、ここ2年くらいは、もっぱらSKKを使っていたが、最近、cannaに移行した。その原因は、SKKを使っていると、どうも小指が痺れるような症状が現れ始めたためである(SKKの入力はSHIFTキーを多用する)。cannaを使ってみて、まず、素の状態のcannaではとても使えないというのが第1印象だった。ただ、UNIXの世界で、それなりに長く使われているソフトウェアだけあって、種々の強化やカスタマイズをすれば、かなり使えるものになる。この文書は、そのメモをまとめたものである。

本文中で、

```
% ps uxw | grep kinput2
% grep XMODIF ~/.??*
# /usr/local/etc/rc.d/canna.sh stop
```

のように、%や#で始まっている行は、UNIXでのシェルコマンドで、%は一般ユーザーでのコマンド、#は管理者(root)でのコマンドを表す。%や#は、シェルのプロンプトを表しているので、入力する必要はない。また、~/という表記は、ユーザーのhome directoryを表している。

\と¥は、同じに見えるかもしれない。これは、日本語の全角文字で表すと、バックスラッシュ(＼)と円マーク(￥)であるが、画面に表示するフォン

トによっては同じに見えてしまう。また、キーボードによっては、どちらかしかないが、意味は同じである。

この文書の PDF 版は、こちらにあります。

間違い等を見つけたら、ohba@stex.phys.tohoku.ac.jp まで知らせて下さい。

目 次

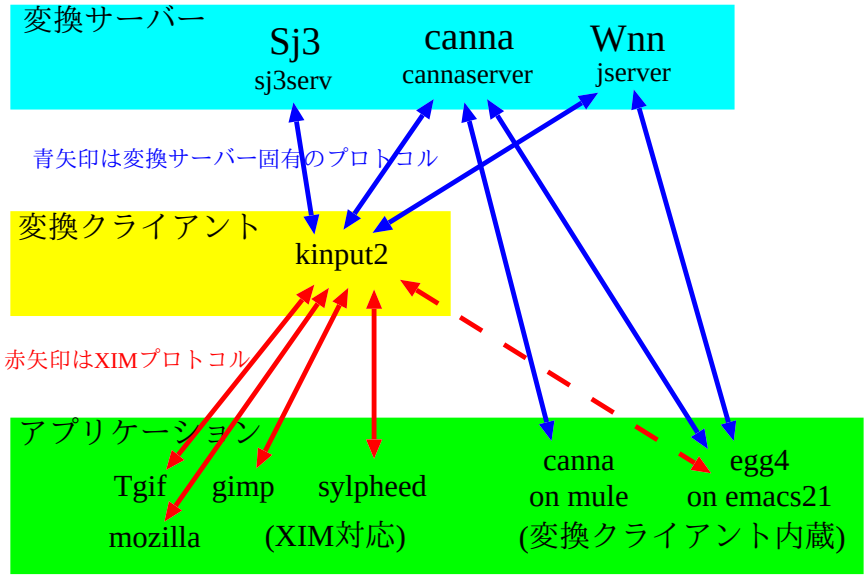

図 1: UNIX における日本語入力の概略図

1 UNIX での日本語入力の仕組み

UNIX 環境では、変換サーバー/変換クライアント/各アプリケーションの 3 者が日本語入力に関係していて、変換サーバーと変換クライアントは、別なマシンにあってもよい。これは、クライアント機が変わっても、同じ辞書を参照できるようにするための仕組みである。これら 3 者のおおまかな関係を簡単に図示すると、次のようになる。この図は、かなり簡略化しており、もっと詳細な関係は、こちらで見ることが出来る。

変換サーバー

変換クライアントから「かな」などを受取り、「漢字」に変換して返す役割を持つ。Wnn [1] の jserver、Canna [2] の cannaserver など。辞書の豊富さ、変換のアルゴリズム、学習機能などが変換効率に影響する。

変換クライアント

ユーザーの入力を受け、変換サーバと通信して漢字にしてアプリケーションに渡す役割。ユーザーインタフェースを受け持ち，使い勝手に大

[1] 京大とオムロンとの共同開発。Version4 までは、フリーソフトウェアとして公開されていたが、現在は、Wnn6, Wnn7 は商用で市販されている。私の名前は中野です、を一発で変換できるようにするのを目標として開発され、それが Wnn の語源。

[2] 元々は NEC で開発され、現在は、sourceforge.jp 上で開発が続けられている。

きな影響を持つ。kinput2、 xwnmo[3]、uum[4]、canuum[5]、 skkinput[6]など。

アプリケーション

X[7]上の国際化したアプリケーションは、変換クライアントと通信する能力をもつ (tgif,mozilla など)。また、emacs のように、アプリケーションによっては、自前で変換クライアント機能 (たまご[8]や canna) を持ち、直接、変換サーバーと通信するものもある。

変換サーバーと変換クライアントとの関係

どの変換サーバーと通信できるかは、変換クライアントによって決まっている。uum や xwnmo は Wnn と、canuum は Canna と通信できる。kinput2 は、コンパイル時の指定で、Canna、Wnn(FreeWnn, Wnn6, Wnn7)、Sj3 と通信可能になる。

変換クライアントとアプリケーションとの関係

X での変換クライアントとアプリケーションが通信する方法には、歴史的にいくつかのプロトコルが用いられて来た。例えば、変換クライアントである kinput2 は、manual page によれば、

```
kinput protocol
kinput2 protocol
Matsushita jinput protocol
Sony xlc protocol
XIMP protocol
X Input Method Protocol (X Consortium standard)
```

のプロトコルに対応している。また、X でのお絵書きソフトである tgif は、”xcin”, ”chinput”,”kinput2”, ”xim” の４つのプロトコルに対応している。現在は、XIM(X Input Method) プロトコルがほぼ標準になってきたので、国際化したほとんどのアプリケーションは、XIM に対応していると考えて間違いない。XIM で実際に使う変換クライアントは、通常、環境変数 XMODIFIERS で指定する。

1.1 日本語入力の手順

一般的に、日本語入力は次の手順で行われる。

---

[3]Wnn 用の X 上の変換クライアント。
[4]Wnn 用の文字端末用変換クライアント。wnn を 180 °回転させた。
[5]Canna 用の uum
[6]SKK 用の X 上の変換クライアント
[7]X Window system のこと。略称は、X または、X11 で、X window とは呼ばない。
[8]Wnn の変換クライアント。**た**くさん**ま**たせて**ご** めんなさい、の頭文字をとった。その英訳で egg とも呼ばれる。

- 通常のアルファベット入力 (入力したキーボードの文字がそのまま出力される状態) から、ある特別なキー (**変換起動キー**と呼ぶことにする) を押すと、日本語入力が ON になる。変換起動キーは変換クライアントに依存するが、kinput2 では SHIFT + Space または Ctrl + o (おー)、tamago4 では Ctrl + \ に割り当てられている。

- 日本語入力が ON になると、キーボードからの入力を、変換クライアントが一旦奪って、かな や アルファベット に変換してから、画面に表示する。この状態を**フェンスモード**と呼ぶ。

- フェンスモードで Space キーを押すと、変換クライアントは、フェンス内の文字列を変換サーバーに送る。変換サーバーは、受け取った文字列を、文節を区切るなどして漢字に変換し、結果を変換クライアントに返し、変換クライアントは結果を画面に表示する。この状態を、**漢字変換モード**または単に**変換モード**と呼ぶ。漢字変換モードでは、文節毎に他の候補を選んだり、文節を伸ばしたり縮めたりしながら、目的とする日本語にインタラクティブに変換する。漢字変換モードで、Return キーを押すと、文字列を確定して、日本語入力 ON の状態に戻る。

- 日本語入力が ON の状態である特別なキー (**変換終了キー**と呼ぶことにする) を押すと、日本語入力が OFF になり、アルファベット入力に戻る。通常、変換終了キーは、変換起動キーと同じ場合が多い。

以上の操作のように、ユーザーの側から見ると、日本語入力システムはひとつに見えるかもしれないが、内部的には、変換クライアントと変換サーバーの役割は明確に分離している。例えば、「ぁぃぅぇぉ] という「かな] を出したり、”ABC” と入力したとき、”ABC” 、 ”ＡＢＣ”、”あＢＣ”、”あ BC” のどれをフェンスモードに表示するかは、変換クライアントの仕事であり、その部分を使い易くするためには、変換クライアントのカスタマイズが必要である。一方、ある単語が漢字に変換できなかったり、自分で良く使う単語が候補の最後の方に現れるのは、変換サーバーの所為であり、変換サーバーの辞書を賢くしたり、変換サーバーに頻度学習を行わせる必要がある。

2　canna を使う最低限の設定と使い方

canna をいくつかのアプリケーションから使う時の最低限の設定を書く。図に示したように、canna を使う際に、変換クライアント (kinput2) を経由 (その際、XIM プロトコルで通信) して変換を行うものと、自前で変換クライアント機能を内蔵しているもの (mule や emacs) に大別できる。

2.1 kinput2 を経由した日本語入力

kterm, tgif, mozilla, sylpheed, gimp などの X 上のアプリケーションは、変換クライアントである kinput2 経由で、日本語入力をおこなう。この時、通信のプロトコルとして、XIM プロトコルを用いる。

2.1.1 kinput2 の起動

通常は、X から login した場合に実行される ~/.xsession で起動する。kinput2 が起動しているかどうかは、

```
% ps uxw | grep kinput2
```

で確認することが出来る。何も表示されない場合は、

```
% kinput2 -canna -cs cannaserver のホスト名 &
```

と手動で立ち上げるか、~/.xsession に書いて login し直す。ここで、-canna は、変換サーバーとして canna サーバーと通信することを表し、そのサーバーのホスト名を -cs オプションで与える。kinput2 がどの変換サーバーと通信できるかはコンパイル時のオプションで決まっていて、

```
% kinput2 -version
```

で確認できる。その結果、例えば、

```
kinput2 version 3.1  (2002/10/03)
   options: [Wnn] [Canna2]
```

であれば、Canna と Wnn に対応しているので、-canna は必須だが、

```
   options: [Canna2]
```

だけであれば、Canna としか通信できないので、-canna は不要である。

また、cannaserver のホスト名は、stex の場合は、s1 を指定する。なお、環境変数 CANNAHOST が設定されていれば、それを読むので、-cs の指定はなくてもよい。kinput2 を立ち上げたときに、変換サーバーと通信できない場合には、

```
Warning: かな漢字変換サーバと通信できません
```

というメッセージを出し、実際にアプリケーションで、”かな” から” 漢字” に変換しようとしたときに、同じメッセージを出す。この時は、

```
% cannacheck -v -cs cannaserver のホスト名
```

で、cannaserver が立ち上がっているか確認し、立ち上がっていない場合には、管理者に連絡して、立ち上げてもらう。

2.1.2 環境変数の指定と日本語入力の ON/OFF

通常、XIM プロトコルに対応した国際化したアプリケーションは、環境変数 XMODIFIERS を見て、変換クライアントを探す。そこでは、

```
LANG=ja_JP.eucJP
XMODIFIERS=@im=kinput2
```

のふたつの環境変数が関係する。現在設定されている環境変数は (csh,tcsh の場合)、

```
% printenv
```

で見ることができる。ふたつの環境変数が設定されていないときは、

```
% setenv LANG ja_JP.eucJP
% setenv XMODIFIERS "@im=kinput2"
```

と入力して設定するか、または、これらを /.cshrc に書いて、

```
% source ~/.cshrc
```

で、再度読み込む。

各アプリケーションで、日本語入力の起動

以上が正しく設定されていれば、**日本語を入力したい場所で、Shift キーを押しながら Space キーを押す**と、日本語入力モードになる。何も起こらない場合は設定を見直す。アルファベット入力モードに戻るには、再度、Shift + Space を押す。また、Ctrl + o (オー) でも、同様な動作になる。

2.1.3 うまく行かない場合

上記の設定だけでうまく行かない場合は、いくつかの可能性が考えられる。

XIM に対応していないアプリケーション

国際化していないアプリケーション (英語だけにしか対応していないもの) では、日本語入力はできない。例えば、現在のところ、UNIX 上の Acrobat Reader (acroread) は、英語版しか配布されていないので、検索窓で、日本語入力はできない。

XIM だけでなく複数のプロトコルに対応しているアプリケーション

XIM プロトコルが標準となったのは、比較的最近であるため、最近開発されたアプリケーションは XIM だけに対応しているものも多いが、逆に、古くから開発が行われてきたものの中には、XIM だけでなく、他のプロトコルにも対応していて、そのためにうまくいかないことがある。

XIM に対応していない場合はあきらめるしかないが、複数のプロトコルに対応している場合には、設定を正しく行えば、日本語入力が可能になる。ただし、その設定はアプリケーション毎に異なるので、それぞれのアプリケーションの manual page や 開発元の WEB page や インターネットで検索して設定するしかない。また、周りに詳しい人がいれば、その人に聞くのが早道である。一例として、お絵書きソフト tgif の場合について、関係する設定を Appendix に書いておく。

2.2 mule から canna を使う

mule では、変換クライアント機能を内蔵した canna(canna.el)[9] を利用できるので、 kinput2 を必要としない[10]。必要最低限の設定は、~/.emacs ファイルの

```
(load-library "canna")
(setq canna-server "canna サーバーのホスト名")
(canna)
```

の 3 行である。ここで、canna サーバーのホスト名は、stex の場合は、s1 とする。Ctrl + o (おー) で、日本語入力の ON/OFF を行う。

```
(global-set-key "\C-\ " 'canna-toggle-japanese-mode)
```

を加えると、Ctrl + \ でも、日本語入力の ON/OFF が行えるようになる。

2.3 emacs(21.X) から、canna を使う

普通に作成した emacs21[11] は、XIM に対応しているので、X 上で立ち上げた emacs の場合、kinput2 経由での日本語入力が可能である。kinput2 を経由した日本語入力で述べたように、Shift + Space (または、Ctlr + o) で日本語入力の ON/OFF が行える。ただし、この方法では、X 環境以外 (例えば、Windows 機から login している場合) では日本語入力が出来ない。また、以下に説明する emacs 自体の変換クライアント機能は、強力なカスタマイズができるので、kinput2 による日本語入力ではなく、emacs 自体の変換クライアント機能を使うことを強く勧める。

このセクションで説明する emacs 自体の変換クライアント機能を使う場合には、XIM 経由での日本語入力機能は不必要であるだけでなく、邪魔になる場合がある。例えば、「漢字変換モード] で、変換対象の文節を伸ばそうとして、Ctrl+o を押すと、XIM の ON/OFF になってしまい、文節を伸ばすこと

[9] 以下、mule 上で動く canna を canna/mule と表記する。
[10] mule のベースである emacs 19 は、XIM プロトコルに対応していないので、kinput2 と通信できない。
[11] コンパイル時の指定で、without_xim を指定すると、XIM 非対応の emacs が作成される

ができない。XIM 機能を使わなくするためには、環境変数 XMODIFIERS を未設定にすれば良いのだが、シェル環境全体でこのように設定してしまうと、他の X アプリケーションからも、日本語入力ができなくなってしまうので、emacs の場合だけに、そうするには、~/.cshrc に

alias emacs env XMODIFIERS=@im=none emacs

と書いて、login し直すか、すぐに反映させるためには、

```
% source ~/.cshrc
```

で、設定を再読み込みする。

emacs 20 以降では、mule の機能の多くは、emacs 本体に統合されたが、canna.el や たまご (Wnn の emacs クライアント) の機能は、含まれていない。kinput2 に頼らず、emacs 単独で日本語入力環境を整えるために、いくつかの方法が行われている。

Emcws

emacs に パッチを当てて、mule と同じ様に、Canna,Wnn,Sj3 サーバーと通信できるようにする。Emacs + Canna + Wnn + Sj3 の頭文字をとって、Emcws と呼ばれる。

http://emacs-20.ki.nu/emcws.shtml

tamago Ver.4

emacs 本体はそのままで、emacs lisp だけで、Canna,Wnn,Sj3 サーバーと通信できるようにする。たまご は、元々、Wnn に対する Emacs 上の変換クライアントの名前であり、tamago Ver.4 は、たまごのインターフェースで、変換サーバーとしては、Canna や Sj3 も使うことができる。

http://www.m17n.org/tamago/index.ja.html

yc.el

emacs lisp だけで、Canna サーバーと通信する変換クライアント。boiled-egg の機能もあるのが特徴。

http://www.ceres.dti.ne.jp/~knak/yc.html

これらの他にも、いくつかの方法がある ようであるが、 ここでは、Mule プロジェクトの後継のひとつとして開発されている tamagoVer.4[12]を用いた方法を述べる。 以後の記述は、FreeBSD の ports でインストールした tamago Ver.4 を対象にしている。現在、ports でインストールすると、tamago-4.0.6-20011028.patch というパッチが当たったものが作られるので、元々の tamago4 とは、少し違うかも知れない。

tamago Ver.4 がインストールされているかは、システムの emacs lisp directory に egg directory があるかどうかで判断できる。FreeBSD 4.X 上の emacs 21.3 の場合は、

[12]以後、この emacs 上の tamago Ver.4 を tamago4/emacs と表記する。

表 1: 変換起動キーと変換終了キー (括弧内は、キーにバインドした場合)

kinput2	canna/mule	tamago4/emacs
Shift+Space, C-o	C-o, (C-\)	C-\, (C-o)

```
/usr/local/share/emacs/21.3/site-lisp/egg/
```

という directory があれば、tamago Ver.4 が使えると考えて良い。FreeBSD の ports でインストールする場合は、/usr/ports/editors/tamago。システムに、tamago Ver.4 がインストールされていれば、~/.emacs または ~/.emacs.el [13] に次の記述を加えれば、canna が使えるようになる.

```
(setq default-input-method "japanese-egg-canna")
(setq canna-hostname "canna サーバーのホスト名")
```

ここで、canna サーバーのホスト名は、stex の場合は、s1 とする。Ctrl + \ で、日本語入力の ON/OFF を行う (たまごの標準キー)。mule や kinput2 の場合のように、Ctrl + o でも、ON/OFF を行うためには、

```
(global-set-key "\C-o" 'toggle-input-method)
```

を加える。

2.4 kinput2 と canna/mule と tamago4/emacs でのキーバインド

これらは、基本的な操作には、大きな差はない。また、キーバインドは、設定ファイルでカスタマイズ可能である。以下では、ローマ字入力で「ひらがな」が表示されている状態を**フェンスモード**、Space キーを押して文節を分割し、漢字に変換された状態を**漢字変換モード**と呼ぶ。表 1〜3 に、変換起動 (終了) キー、フェンスモードでのキーバインド、漢字変換モードでのキーバインドを示す。C-x は、Ctrl キーを押しながら x キーを押す操作を表し、ESC-x は、ESC キーを押した後で x キーを押すことを表す。

3 cannaによる日本語入力のカスタマイズ法(基本編)

3.1 ユーザー辞書、個人別頻度ファイルの作成

まず、自分専用の辞書 (単語登録した際に登録される辞書) を作成する。

```
% cannacheck -v cannaserver のホスト名
```

[13] emacs20 以降では、~/.emacs.el が存在すれば、それが読まれ、存在しなければ、~/.emacs が読まれる。

表 2: フェンスモードでのキーバインド

機能	kinput2	canna/mule	tamago4/emacs
漢字変換モードへ	Space	Space	Space, C-w
現在のまま確定	Return,C-m	Return,C-m	Return,C-m
左に移動	←, C-b	←, C-b	←, C-b
右に移動	→, C-f	→, C-f	→, C-f
左端へ移動	C-a	C-a	C-a
右端へ移動	C-e	C-e	C-e
前の 1 文字削除 *1	BS, C-h	BS, C-h	BS, C-h
カーソル上の 1 文字削除 *1	DEL, C-d	DEL, C-d	DEL, C-d
カーソル以降を削除	C-k	C-k	C-k
フェンス内を消去	C-g	C-g	C-c
字種変換 *2	↑,C-p, ↓,C-n	↑,C-p, ↓,C-n	
全角大文字に *2	C-u	C-u	
全角小文字に *2	C-l	C-l	
canna 拡張モードへ	HELP*3	HELP*4	
ひらがなに			ESC-h
カタカナに			ESC-k
全角文字に *5			ESC->
半角文字に *5			ESC-<
半角英数入力 *6			q
全角英数入力 *6			Q
全角スペースの入力	@@	@@	Q Space

∗ 1 : DEL や BS キーの動作は、.canna や .emacs の設定に依存する。

∗ 2 : この後、↑ (C-p)、↓ (C-n) で、ひらがな ←→ カタカナ←→半角カタカナ ←→全角英数 ←→半角英数を循環する。canna のこのモードを、**字種変換モード**と言う。

∗ 3 : HELP キーのない場合は、.canna に (global-set-key ”\F1” ’extend-mode) と書いておけば、F1 を代用できる。

∗ 4 : HELP キーのない場合は、ESC-x canna-extend-mode を適当なキーにバインドすればよい。

∗ 5 : フェンス内のアルファベットのみが変換され、かなには影響しない。

∗ 6 : 入力後、確定すれば、通常のローマ字入力に戻る。同じフェンス内でローマ字入力に戻るには、C-g。

表 3: 漢字変換モードでのキーバインド

機能	kinput2	canna/mule	tamago4/emacs
現在のまま確定	Return,C-m	Return,C-m	Return,C-m
フェンスモードへ戻る	C-g,BS	C-g,BS	C-c,BS
現文節をフェンスモードへ	C-c	C-c	
文節を伸ばす	C-o	C-o	C-o
文節を縮める	C-i	C-i	C-i
次の候補	Space,C-n, ↓	Space,C-n, ↓	Space,C-n, ↓
前の候補	C-p, ↑	C-p, ↑	C-p, ↑
候補一覧表示	Space2 回,C-w	Space2 回,C-w	ESC-s
左文節に移動	←, C-b	←, C-b	←, C-b
右文節に移動	→, C-f	→, C-f	→, C-f
左端文節へ移動	C-a	C-a	C-a
右端文節へ移動	C-e	C-e	C-e
現文節を全角大文字に *1	C-u	C-u	
現文節を全角小文字に *1	C-l	C-l	
現文節をひらがなに			ESC-h
現文節をカタカナに			ESC-k

∗ 1 : この後、現文節が字種変換モードになる。

として、

単語登録用辞書 "user"を指定しています。

という行が表示されれば、すでに出来ている。user が表示されなければ、

```
% mkdic -cs  cannaserver のホスト名 user
  New dictionary "user" is created.
  Please change customize file.
```

で、作る。ここで、“Please change customize file.” は、~/.canna ファイルに、

```
(use-dictionary
...
:user  "user" ; この行
)
```

があるかどうかで、なければ追加しなさい、の意味。

次に、個人別頻度ファイルの作成をおこなう。大きなシステム辞書については、頻度情報を個人で持った方がよい。そうしないと、他の人が最後に変換した単語が、候補の最初に出てきてしまう。どういうシステム辞書があるかは、

```
% lsdic -a -cs cannaserver のホスト名
```

とすれば、表示される。ただし、ここで表示されるすべての辞書について、個人別頻度ファイルを作成する必要はない (できない)。個人別頻度ファイルが作成できるのは、バイナリー辞書で、システムの頻度ファイルが存在するもの (たぶん、自立語辞書) だけである。実際にその条件を満たすシステム辞書は、cannaserver の動いているホストの、システムの dics.dir を見るとわかる。dics.dir ファイルは、FreeBSD では、/usr/local/share/canna/dic/canna/ (古いシステムでは、/usr/local/lib/canna/dic/canna/なので、以下の記述は、読みかえること) に存在する。

canna では 頻度ファイルは .cld の拡張子なので、cannaserver の動いているホストで、

```
% grep cld /usr/local/share/canna/dic/canna/dics.dir

  gcanna.cld(gcanna.mwd)  -gcanna---
```

として表示される gcanna (右端の - と — で囲まれた部分) について、

```
% mkdic -fq -cs cannaserver のホスト名 gcanna
```

とすれば、システム辞書 gcanna についての、個人別頻度ファイルが作成される。個人別頻度ファイルは、cannaserver の動いているホストの/usr/local/share/canna/dic/user/ユーザー名/ 以下に出来る。/usr/local/share/canna/dic/user/ユーザー名/dics.dir ファイルの中身を見てみよう。

なお、このように 個人別頻度ファイルを作成した後で、システム辞書が変更されると (管理者がより強力な辞書に入れ換えた場合など)、個人別頻度ファイルとシステムの頻度ファイルに矛盾が生じて、そのシステム辞書が使えなくなる。

```
% cannacheck -cs cannaserver のホスト名
```

すると、

```
XXXXX をマウントできませんでした
```

といわれる。この場合には、再度、個人別頻度ファイルを作りなおす必要がある。

```
% rmdic -fq -cs  cannaserver のホスト XXXXX
% mkdic -fq -cs  cannaserver のホスト XXXXX
```

以上の操作は、コマンドに “-cs cannaserver のホスト名” を付けて実行したことからもわかるように、変換サーバーのカスタマイズであり (つまり、「かな」から「漢字」にどのように変換するか)、変換クライアント (kinput2 や canna/mule、tamago4/emacs) とは独立なカスタマイズである。

3.2 変換クライアントのカスタマイズ

キーボードからキーを入力した際に、画面上にどのような文字 (かな) を表示するかは、変換クライアントの仕事である。普通は、ローマ字入力が多いが、カスタマイズすれば、「かな入力」も可能になる。

3.2.1 kinput2, canna/mule のカスタマイズ

これらの変換クライアントは、~/.canna ファイルでカスタマイズを行う。~/.canna ファイルが存在しない場合は、/usr/local/share/canna/default.canna が使われる。

自分の home directory に、.canna が存在しない場合は、/usr/local/share/canna/sample/ directory にカスタマイズ用のサンプルファイルがあるので、ひとつをコピーする。普通は、unix.canna ファイルでよいが、ATOK の入力に慣れている場合には、just.canna の方がよいかもしれない。

```
% cp /usr/local/share/canna/sample/unix.canna ~/.canna
```

~/.canna ファイルでの基本的なカスタマイズのみを書く。詳しいことは、canna のマニュアルのカスタマイズの章を参照。

1. ローマ字かな変換テーブルの指定

   ```
   (setq romkana-table "default.cbp")
   ```

   default.cbp は、/usr/local/share/canna/dic/default.cbp にある。この directory には、just.cbp(ATOK 風)、vje.cbp(VJE 風) などいくつかの変換テーブルがあるので、default.cbp を適当なものに変更すればその変換テーブルに変わる。ただし、このファイル自体はバイナリーファイルで、どういう変換テーブルなのかわからない。変換テーブルのソースファイルは、/usr/local/share/canna/sample/src/ 以下に同じファイル名である (拡張子は .ctd) ので、眺めてみよう。

   自分で、ローマ字かな変換テーブルをカスタマイズするためには、/usr/local/share/canna/sample/src/ のファイルのひとつを持ってきて、編集し、変換テーブルのバイナリーファイルを作成して、home directory に置けば良い。

   ```
   % cp /usr/local/share/canna/sample/src/default.ctd my-table.ctd
   % emacs my-table.ctd (編集)
   % mkromdic my-table.ctd
   ```

   で、my-table.cbp を作製し、~/.canna ファイルを、

```
(setq romkana-table "my-table.cbp")
```

に変更して、kinput2 などを再起動すれば、変更される。

2. 使う辞書の指定

```
;;; 利用する辞書
(use-dictionary
"gcanna"
"fuzokugo"
"hojomwd"
"hojoswd"
:bushu "bushu"
:user  "user"
)
```

の部分で、実際に利用する辞書を定義している。

```
% lsdic -a -cs cannaserver のホスト名
```

で表示される辞書で、ここにないものは、追加しておこう。実際に、使用している辞書は、

```
% cannacheck -v -cs  cannaserver のホスト名
```

で確認できる。

3. extended-mode へのキーバインド (kinput2 のみ)

   canna では Help キーで、拡張モードに入り、単語登録や各種設定、記号入力などが出来る。Help キーのないキーボードもあるので、

```
(global-set-key "\F1" 'extend-mode)
```

   を入れておくと、F1 キーで、拡張モードに入れるようになる。

3.2.2 tamago4/emacs のカスタマイズ

tamago4 は、変換サーバーとして canna も利用できる Wnn の変換クライアントであり、.canna ファイルは読まない。.canna ファイルに対応するのは、.eggrc ファイルであるが、.canna のような細かいカスタマイズは出来ない。~/.eggrc ファイルが存在しない場合は、/usr/local/share/emacs/21.3/site-lisp/egg/eggrc が読まれる。

1. .eggrc ファイル

   ~/.eggrc ファイルでは、使う辞書の指定をする。
   /usr/local/share/emacs/21.3/site-lisp/egg/eggrc には、変換サーバーとして、
   Wnn と canna を使う場合の両方の記述があるが、canna を使う場合には、

```
(canna-define-environment)
;(canna-add-dict "iroha" nil) ; nil は書き込みしない
(canna-add-dict "gcanna" nil)
(canna-add-dict "gcannaf" nil)
 ...
(canna-add-dict "user" t) ; t は書き込みする
```

   ~.canna ファイルの場合と同様に、lsdic -a で表示される辞書は、追加し
   ておこう。

2. ローマ字かな変換テーブルのカスタマイズ

   現在の tamago4 では、kinput2, canna/mule の様に、洗練された方法でのカ
   スタマイズはできない。ローマ字かな変換テーブルは、/usr/local/share/emacs/21.3/site-
   lisp/egg/its/hira.el に書き込まれていて、これを変更するには、~/.emacs
   または~/.emacs.el ファイルで再定義するしかないようである。

   例えば、default のテーブルでは、“lalilulelo” と入力すると “らりるれろ
   “に変換されるが、これを”ぁぃぅぇぉ” にするには、

```
  (defun my-customize-romaji-tamago4-hira ()
(interactive)
        (define-its-state-machine-append
          its-hira-map
            (its-defrule "la" "ぁ" nil t)
            (its-defrule "li" "ぃ" nil t)
            (its-defrule "lu" "ぅ" nil t)
            (its-defrule "le" "ぇ" nil t)
            (its-defrule "lo" "ぉ" nil t)
           )
  )
 (eval-after-load "its/hira" '(my-customize-romaji-tamago4-hira))
```

   を、~.emacs.el(または、~.emacs) に書く。筆者は、記号や数字が全角文
   字ではなくアルファベットで出た方が便利なので、

```
            (its-defrule "(" "(" nil t)
```

```
                (its-defrule ")" ")" nil t)
                (its-defrule ":" ":" nil t)
                (its-defrule "*" "*" nil t)
                (its-defrule "!" "!" nil t)
                (its-defrule "%" "%" nil t)
(its-defrule "\\" "\\" nil t)
(its-defrule "@" "@" nil t)
(its-defrule "\"" "\"" nil t)
(its-defrule "1" "1" nil t)
                (its-defrule "1" "1" nil t)
                (its-defrule "2" "2" nil t)
                (its-defrule "3" "3" nil t)
                (its-defrule "4" "4" nil t)
                (its-defrule "5" "5" nil t)
                (its-defrule "6" "6" nil t)
                (its-defrule "7" "7" nil t)
                (its-defrule "8" "8" nil t)
                (its-defrule "9" "9" nil t)
                (its-defrule "0" "0" nil t)
```

という記述 (実際にはもう少し多い) も、上記の its-hira-map に加えている。このように設定した後で、全角の記号や数字を入力したい場合には、”Z” の後に、その記号を入力すればよい。余談だが、tamago では、小文字の ”z” の後にキーを押すと、色々な記号がでるようになっているので、試してみると面白い。

3. その他のカスタマイズ
   すべて、~.emacs.el(または ~.emacs) に記述する。

   - 変換候補の一覧を自動的に出す

     canna を使っていて、tamago4 に乗り換えると戸惑うのが、変換候補のリストが自動的に出ないことである。変換モードで、ESC-s で出すことができるが、自動的に出すようにするには、

     ```
     (setq egg-conversion-auto-candidate-menu 1)
     ```

     とする。0 にすると、候補一覧を表示しない。

   - 候補選択時，数字あるいはアルファベットを入力した時点で候補を確定にする

     ```
     (setq menudiag-select-without-return t)
     ```

もっと色々とカスタマイズ可能なので、この辺を参考にしてください。

http://sodan.org/~knagano/emacs/tamagov4.html
http://phe.phyas.aichi-edu.ac.jp/~cyamauch/pl2_emacs21.html

3.3 canna のユーザー辞書への単語登録

変換できなかった単語や、単語だけでなく頻繁に使う長い文章などはユーザー辞書に登録しておくと、以降の変換効率が上がるので、できるだけ登録しよう。

kinput2 から

HELP キー または F1 キー を押して出る 「拡張メニュー」から、「４単語登録」を選ぶ。

単語登録したいということは、その単語が変換されなかったということで、その単語を出すために、すでに、何らかの操作をしているはず。kinput2 から登録する場合は、その操作をもう一度する必要があるので、やや不便。

なお、途中で「拡張メニュー」から抜けるためには、C-g を押す。

canna/mule から

登録したい単語をリージョンに入れて、ESC-x canna-touroku-region。

再度、単語を変換する必要はない。「リージョン」とは、マークした位置から現在のカーソルの位置までの領域を指す。。マークするためには、C-Space。X の場合は、マウスの左クリックしながら、領域を選ぶことでも、リージョンに入れることができる。なお、ESC-x の後は、TAB で補完ができる。候補が複数ある場合は、再度 TAB を押すと、候補一覧がでる。

tamago4/emacs から

登録したい単語をリージョンに入れて、ESC-x egg-toroku-region。

単語登録を、キーにバインドする

ESC-x egg-toroku-region などの操作が面倒であれば、キーにバインドしておこう。例えば、F7 にバインドするためには、canna/mule では、~/.emacs に

```
(global-set-key [f7] 'canna-touroku-region) ;; F7 で touroku-region
```

tamago4/emacs では、~/.emacs.el に

```
(global-set-key [f7] 'egg-toroku-region) ;; F7 で toroku-region
```

と書く。

3.4 コマンドラインでのユーザー辞書の操作

登録されている単語の一覧表示

user 辞書に登録されている単語の一覧は、

```
% catdic -cs cannaserver のホスト名 user
```

で得られる。出力としては、例えば、

```
FreeBSD #T30 http://www.jp.freebsd.org/
こぴぺ #T35 コピー&ペースト
そふとうぇあ #T35 ソフトウェア
どしー #T35 \\textcelsius{}
みかみね #CN 三神峯
おろかもの #T35 悪露化者
よろしく #T35 宜しくお願い致します。
```

のように、読み 品詞コード 漢字 (でなくてもよい) になっている (辞書ファイルの形式参照)。。

登録されている単語の削除

間違って登録してしまった単語を削除するには、上記の catdic の出力をそのまま、コピー&ペーストして、

```
% echo 'おろかもの #T35 悪露化者' | delwords -cs ホスト名 user
```

とする。

単語の登録

削除の delwords の代わりに、addwords にして、

```
% echo 'おろかもの #T35 愚か者' | addwords -cs ホスト名 user
```

とする。

空白を含む単語の登録

例えば、"とんぺい" と入力して、"Tohoku University" と変換するためには、Space の 前に、\ を入れて[14]、

[14]canna の辞書の形式では、Space は 読み 品詞 漢字 1 漢字 2 の並びの区切り文字として解釈されてしまう。このように、ある意味をもった文字の機能をキャンセルし、一般文字としてソフトウェアに認識させることを"**エスケープする**"と呼び、そのための文字 (**エスケープキャラクター**) として、UNIX では通常 \ が用いられる。

```
% echo 'とんぺい #T35 Tohoku\ University' | addwords -cs ホスト名 user
```

で OK。

特殊文字を含む単語の登録

例えば、TeX で良く使う \textbackslash{} を"\\" と単語登録するためには、\ は、特殊記号なので、空白の場合と同様に、\ でエスケープして、

```
% echo '\\\\ #T35 \\textbackslash{}' | addwords -cs ホスト名 user
```

で OK。

単語の一括登録と削除

フリーな辞書をダウンロードして、個人の辞書ファイルに登録するには、canna に付属する、addwords と delwords コマンドを使う。例えば、user 辞書に、hoge.t ファイルを一括登録するためには、

```
% addwords -cs cannaserver のホスト名 -l hoge.t user
```

でよい。逆に、辞書から一括削除するためには、

```
% delwords -cs cannaserver のホスト名 user < hoge.t
```

で OK。

4 canna 辞書の強化

「canna はお馬鹿さん」と良く言われるが、それは、canna 標準の辞書が貧弱であることの原因が大きい。元々の canna に付属するシステム辞書 (iroha) は、約 4 万語しか単語を持っていない。iroha では、例えば、「試料」「分光」「分光器」「異方性」「等方性」などの単語もないため、これを使って、分光学的な日本語文書を書くことは、不可能に近い。

> 市販の日本語入力ソフトに付随する辞書の単語数は、例えば、Macintosh で古くから使われてきた EGbridge の場合、
> http://www4.airnet.ne.jp/koabe/com_inet/im/feature.html によれば、「メイン辞書＝約 27 万語、地名人名辞書＝ 28 万 3 千語、医学辞書＝ 15 万 4 千語」らしい。

canna 辞書を強化するプロジェクトが進行中で、現在 (2004.1)、14 万語が収録されている。iroha 辞書を、この成果物である、gcanna 辞書に入れ換えるだけで、上記の、「試料」... などの５つの単語が変換可能になる。更に、イン

ターネット上には、多くのフリーのかな漢字変換辞書があるので、それらを利用することで、canna の変換効率を上げることができる。

ここで用いているスクリプトプログラムなどは、配布パッケージ
http://www.stex.phys.tohoku.ac.jp/~ohba/misc/canna-util.tar.gz
にまとめてある。

4.1 強化した canna 辞書のインストール法

FreeBSD では、ports の japanese/cannadic をインストールすればよい。インストール後、

```
/usr/local/share/doc/cannadic/README.FreeBSD
```

の指示にしたがって、dics.dir ファイルの変更などを行う必要がある。この辺りは、どのような辞書の場合も同じなので、次節の一般的な辞書ファイルの追加の方法を参照。

4.2 辞書ファイルのシステム辞書への追加方法

1. テキスト辞書から、バイナリー辞書に変換

   ダウンロードした canna 用の辞書ファイルが、テキスト形式の場合は、バイナリー辞書に変換する。テキスト形式の canna 用辞書は、*.t または、*.ctd の名前である。

   ```
   % mkbindic hoge.ctd (または、hoge.t)
   ```

   これで、hoge.cbd(バイナリー辞書本体) と hoge.cld(頻度ファイル) ができる。

2. バイナリー辞書のシステム辞書への追加

   作成したバイナリー辞書を、canna の辞書 directory にコピーして、dics.dir ファイルに追加した辞書の情報を書き加える。これには、管理者権限が必要。

   ```
   # cp foge.cbd /usr/local/share/canna/dic/canna/
   # cp foge.cld /usr/local/share/canna/dic/canna/
   # vi /usr/local/share/canna/dic/canna/dics.dir
   ```

   dics.dir は、

   辞書のファイル名 (その辞書の中の辞書名)　-辞書名---

の形式である。canna のバイナリー辞書は、ひとつのファイルに複数の辞書を持つことができるので、() 内の情報は必要。テキスト辞書は、ひとつの辞書しか持てないので、() 内の名前は省略できる。() 内の辞書の拡張子は、自立語辞書は .mwd、付属語辞書は .swd になる。最後の -辞書名— は、クライアントから見たときの辞書名 (lsdic したときに見える) である。また、バイナリーファイルについては、辞書本体 (.cbd) と頻度情報ファイル (.cld) の両方を書いておく。頻度ファイルを書かないと、個人用頻度ファイルの作成 (mkdic -fq) ができないようだ。dics.dir ファイルの

```
fuzokugo.cbd(fuzokugo.swd)      -fuzokugo---
iroha.cbd(iroha.mwd)            -iroha---
iroha.cld(iroha.mwd)            -iroha---
iroha.cbd(bushu.mwd)            -bushu---
bushu.cld(bushu.mwd)            -bushu---
hojoswd.ctd(.swd)               -hojoswd---
....
```

ここに、mkbindic でできたふたつのファイル (辞書本体と頻度ファイル) に付いての記述、

```
hoge.cbd(hoge.mwd)              -hoge---
hoge.cld(hoge.mwd)              -hoge---
```

を追加すれば良い。なお、canna の辞書ファイルのファイル名は、拡張子を除いて、アルファベット 9 文字までしか使えないようだ。それよりも長い名前を用いてもエラーは出さないが、lsdic しても現れないので注意が必要。

以上の後で、cannaserver をリスタートする。

```
# /usr/local/etc/rc.d/canna.sh stop
# /usr/local/etc/rc.d/canna.sh start
```

3. 個人の設定ファイルの変更

以上の操作で、cannaserver は新しい辞書を使えるようになったが、実際にどの辞書を使うかは、ユーザー毎に選択可能である。一般ユーザーで、

`% lsdic -a -cs cannaserver` のホスト名

で、新しく追加した辞書が見えるようになったはずである。これを使えるようにするには、kinput2 と canna/mule では、 /.canna ファイルに

```
(use-dictionary
  "fuzokugo"
  "hojomwd"
  ...
  "hoge" ; <= 追加
  :bushu "bushu"
  :user  "user"
)
```

を追加し、kinput2(または mule) を再起動。

tagago4/emacs では、 /.eggrc ファイルに

```
(canna-add-dict "hoge" nil)
```

を加えれば良い。

4.3 フリーな辞書たち

以下の辞書には、canna 用だけでなく、その他の形式の辞書も含まれている。ダウンロードした辞書が、テキスト形式のものか、何らかの方法でテキスト形式に変換できれば、canna 形式の辞書に変換することができる。他の形式の辞書から、canna 形式に変換するためには、各種辞書の形式、ATOK 辞書の canna 辞書への変換 や SKK 辞書の canna 辞書への変換 を参照。また、UNIX 上のフリーの日本語入力システムのもう一つの巨頭である Wnn の辞書は、canna 辞書との間で相互に変換可能になっている (Appendix 参照)。

以下の辞書は基本的にはフリーなものであるが、再配布について条件を付けているものもあるので、それぞれの README ファイル等を参照すること。

郵便番号辞書

http://bonobo.gnome.gr.jp/~nakai/canna/

http://bonobo.gnome.gr.jp/~nakai/canna/zipcode-20021021.tar.bz2

これを入れると、“９８０８５７８” という入力が、“宮城県仙台市青葉区荒巻字青葉 2-1 東北大学　(理学部・薬学部等)“ と変換できるようになる。

物理用語集 (AOK8 用)

http://www.vector.co.jp/soft/data/writing/se039662.html

登録単語数: 9862

理化学辞書 5.2(ATOK8 用)

http://lsd.pharm.kyoto-u.ac.jp/dic/others/Chiku.html

登録単語数: 19443

ライフサイエンス辞書プロジェクト (Lsd3,canna 用)

http://lsd.pharm.kyoto-u.ac.jp/index-J.html

登録単語数: 28691

天文・天文物理用語辞書 (Wnn/canna 用)

http://www.kwasan.kyoto-u.ac.jp/~tanuma/tanudic.html

登録単語数: 3418

地球物理辞書 (Wnn,canna,ATOK,MS-IME)

http://www.chibutsu.org/jisho/

登録単語数: 5264

岩石学辞書 (canna 用)

http://www.vector.co.jp/soft/dl/data/writing/se125795.html

登録単語数: 1252

skk 辞書

http://openlab.ring.gr.jp/skk/dic-ja.html

登録単語数: 約 18 万語 (どんどん増えている)

Canada 式 医学用語変換辞書

http://spica.onh.go.jp/med_dic/

登録単語数: 約 10 万語

この他にも、http://www.vector.co.jp/vpack/filearea/data/writing/dic/index.html には、多くの辞書がある。

4.4 各種形式辞書の canna 辞書への変換

いくつかのテキスト辞書の形式を以下に示す。

canna 辞書

読み 品詞のコード 単語 の順に並んでいる (区切り文字はスペース)。

例)

あーべるへんかん `#T35` アーベル変換

あぼがどろ `#JN` アボガドロ

ATOK 辞書 (S-JIS 漢字コード)

読み、単語、品詞名 の順に並んでいる (区切り文字は、コンマ)。

例)
あーべるへんかん、アーベル変換、一般名詞
あぼがどろ、アボガドロ、固有人名

SKK 辞書

読み /漢字 1/漢字 2/... と、一つの読みに対し、複数の漢字を書ける。品詞の概念は基本的にはない。ただし、動詞や形容動詞など、送り仮名が活用で変化するものに付いては、最後は子音のアルファベットで終わっている。

例)
いじょう /以上/異常/≧/
と k /溶/解/[け/溶/]/[き/解/]/

したがって、ATOK 辞書を canna 辞書に変換するためには、ATOK の品詞名を canna の品詞コードに変換し、並び順を変え、区切り文字をコンマからスペースに変えればよい。また、SKK 辞書を変換するためには、読みと漢字を 1:1 にしてから、動詞や形容動詞 (アルファベットで終わっている読み) を取り除き、残ったものを名詞として、並べればよい。以下で、変換のために用いているスクリプトは、配布パッケージに含まれている。

参考 ATOK や canna の品詞情報

http://hp.vector.co.jp/authors/VA014135/fep2.htm
http://www.tanu.org/~sakane/doc/public/howto-canna.html

4.4.1 ATOK 辞書の canna 辞書への変換

フリーな辞書たちの物理用語集 (physics.lzh) を例にして、手順を示す。

1. 辞書のダウンロードと解凍

   http://www.vector.co.jp/soft/data/writing/se039662.html

   から、ダウンロードした physics.lzh を解凍する。

   ```
   % lha -x physics.lzh
   ```

   これでできた、physics.txt が、変換したいテキスト辞書である。

2. 辞書の変換

```
% nkf -e < physics.txt > p.t
```

まず、nkf -e を通して、EUC コードに変換。これで、半角カナは全角カナになり、区切り文字は、”、”になった。 次に品詞を変換して、canna 形式にする。

```
% perl atok2canna.pl < p.t > physics.t
```

これでできた physics.t を mkbindic にかければ、システムバイナリー辞書の出来上がり。

4.4.2 skk 辞書の canna 辞書への変換

1. skk 辞書を list 形式に

   skk 辞書の、

   よみ /漢字１/漢字２/...

   の形式を、

   よみ 漢字１
   よみ 漢字２
   ...

   の形式の変換する。

   ```
   % skk2list SKK-JISYO.L > skk.1
   ```

   skk2list は、skk のパッケージに含まれている perl スクリプト。

2. skk の動詞や形容動詞の単語などを除く。

   動詞や形容動詞を表す、

   わる s 悪
   わる k 悪

   や、接頭語や接尾語を表すと思われる、

   こ> 小
   こ> 故

   >あい 愛
   >あじ 味

の単語を除く。

```
% meishi < skk.1 > skk.2
```

3. 残った単語を、名詞として、canna 辞書に変換する。

   残った単語の中には、人名や地名などが含まれているが、それらを区別する方法がないので、すべて「名詞」として、変換する。

```
% list2canna < skk.2 > skk.t
```

上記に使ったスクリプトは、すべて、
配布パッケージに含まれている。

4.5 辞書をまとめる (マージ等の操作)

複数の辞書をシステム辞書に追加していくと、dics.dir は長くなり、lsdic -a したときに表れる辞書名も多くなり、繁雑になる。また、異なった辞書には重複した単語も登録されているので、無駄が多い。そこで、ここでは、複数の辞書をひとつのまとめたり、追加する辞書から、すでに登録されている単語以外の単語だけを抜き出す方法を述べる。

1. バイナリー辞書からテキスト辞書への変換

   以下の操作は、「読み 品詞コード 単語」形式のテキスト辞書を対象としている。バイナリー辞書しかない場合に、テキスト形式辞書に変換する必要がある。必要なプログラムは、すべて、canna に付属しているもので、/usr/local/bin/ 以下にある。

   (a) バイナリー辞書の辞書名の表示

```
% dpbindic gcanna.cbd


dpxdic gcanna.cbd
gcanna.mwd [ Thu Dec 18 19:49:25 2003 ] = 146230 + 90687
```

   これから、gcanna.cbd は、単語数 146230/読みの数 90687 の自立語辞書 (gcanna.mwd) だけからできていることがわかる。

   (b) バイナリー辞書から、テキスト辞書を取り出す。

```
% dpbindic gcanna.cbd  gcanna.mwd > gcanna.1
```

   これで、テキスト辞書が、gcanna.1 にできる。

(c) 1 行 1 単語に変換する。

このファイル (gcanna.1) は、

```
読み 品詞コード 単語 1 単語 2 ...
```

となっているので、1 行 1 単語に変換し、ついでに重複を除く。

```
% splitword gcanna.1 | sort | uniq > gcanna.t
```

2. 複数の辞書をまとめてひとつにする。

```
% merge_tdic.pl dic.1 dic.2 dic.3 ... | sort | uniq > all.t
```

dic1, dic2, ... は、「読み 品詞コード 単語」形式のテキストファイルに形に変換しておく。ただし、skk 辞書は、人名も地名もすべて名詞として登録されているので、それらは、sort — uniq しても、重複は除けない。

3. 登録されていない単語だけを抜き出す。

単純に辞書を追加していくと、すでに登録されているものが、重複して登録されてしまう。そのためには、登録されていないものだけを抜き出す操作が必要。GNU 版 diff を使えば、ふたつのファイルに対して、最初のファイルだけにある行 (–old-line-format)、ふたつめだけにある行 (–new-line-format)、両方にある行 (–unchanged-line-format) を、それぞれ出力することが簡単にできる。

cannadic プロジェクトの gcanna.t(146230 単語) と skk.t(162543 単語) に対して、

```
% diff  --old-line-format="%L" --new-line-format=""\
  --unchanged-line-format="" gcanna.t skk.t > only-gcanna.t

% diff  --old-line-format="" --new-line-format="%L"\
  --unchanged-line-format="" gcanna.t skk.t > only-skk.t

% diff  --old-line-format="" --new-line-format=""\
  --unchanged-line-format="%L" gcanna.t skk.t > common.t
```

を行うと、

```
only-gcanna.t 106463
onky-skk.t 122706
common.t  39794
```

で、約４万語近くが重複していた (skk 辞書は、すべての単語の品詞を名詞で登録しているので、実際の重複はもっと多い)。gcanna がシステム辞書として、動いている場合は、only-skk.t をバイナリー辞書に変換して登録すれば良い。

5　tamago4/emacs のカスタマイズ (上級編)

この章で述べる emacs 上での tamago4 の設定は、個人的な好みの問題で、必ずしも万人に推奨される設定ではない。また、かなり高度な設定も含まれているので、初心者は行う必要はない。

5.1　load-path の追加

以下のカスタマイズでは、ユーザーが自分専用の emacs lisp プログラムをロードして、カスタマイズを行うので、最初に、その方法を書く。load-path とは、例えば、~/.emacs などで、
(load ”canna”)
という記述があったときに、この canna というプログラム (実際には、canna.elc または canna.el というファイル) を、どの directory から順番に探すかを指定する変数である。emacs を立ち上げて、ESC-x describe-variable [Return] load-path で、確認することができる。

home directory の lisp directory を load-path に加えるためには、~/.emacs.el (ない場合は ~/.emacs) ファイルに、

```
(setq load-path (cons (expand-file-name "~/lisp/") load-path))
```

と書く。これで、~/lisp directory に自分専用の lisp プログラムを置いて、ロードできるようになった。

5.2　日本語での incremental search を可能に

tamago4/emacs では、C-s や C-r での検索の際に、日本語が入力できないバグがある。 tamago-isearch-fix.el を持ってきて、~/lisp directory に置いて、~/.emacs.el ファイルに

```
(load "tamago-isearch-fix") ; incremental search で日本語入力可能に
```

と書けば、OK。これで、日本語での検索が出来ることが確かめられたら、tamago-isearch-fix.el を バイトコンパイル しておこう。

5.3 tamago4 でも canna の機能を使う

tamago4 は、変換サーバーとして、cannaserver を利用できるだけなので、ユーザーインターフェースは、egg そのものである。canna の持っている部首入力や記号入力も使えない。tamago4 でも canna の機能を使えるようにしている人がいる。なお、このページにある、tamago 4.0.6 に対するパッチの多くは、ports でインストールした場合には当たっているようである。egg-canna.el を、~/lisp directory に置いて、~.emacs.el ファイルに、

```
(add-hook 'canna-load-hook '(lambda () (load "egg-canna")))
```

と書くと、canna の部首入力や記号入力が使えるようになる。また、canna-extended-mode もメニューとしてはあるが、機能は使えないようである。

5.4 日本語とアルファベットの混ざった文章をストレスなく入力する

例えば、この文章もそうですが、計算機関係の文章や TeX や html のソースファイルを書く場合には、日本語と英語 (アルファベット) が混ざり、その度に、入力方式を切替えたりするのは、思考の妨げとなる。日本語入力の ON/OFF をしないで、一時的にアルファベット入力に切替える方法も準備されている (skk では "l"、tamago では "q") が、文字を入力する前に余計なキーを押す必要があるので、思考が中断される。canna では、入力した後で、↓ (C-n) や ↑ (C-p) で字種の変換ができるので、 入力する前にキーを押すよりは負担は少ないが、tamago4/emacs では、実装されていない。

日本語とアルファベットが混ざりあった文章をスムーズに入力するために、いくつかの便利な (？) lisp プログラムが公開されている。解説は、こちら。

よく使われるもののひとつは、ゆでたまご (boiled-egg) と呼ばれるもので、基本はアルファベット入力で、「かな」に変換したいところで、変換キー (default は、C-j) を押して、フェンスモードにする。元々は、egg/mule 用に開発され、canna/mule 用 (boiled-canna)、tamago4/emacs 用 boiling-egg) も存在する。

もうひとつよく使われるものは、egg-mix と呼ばれるもので、日本語入力を基本として、アルファベットだと判断した文字列は、かなに変換せずに、アルファベットのまま確定するインターフェースである。egg-mix は egg/mule 用に開発され、tamago4/emacs 用のものは egg-remix と呼ばれる。「ゆでたまご」の系列は、実際には使ったことがないのでわからないが、egg-remix を使ったところ非常に便利で、現在は常用しているので、ここで紹介する。

1. egg-remix のインストール
   こちらから、egg-remix.el を持ってきて、~/lisp/ directory の置き、~/.emacs.el に

```
(setq egg-mode-preference "remix")
(load "egg-remix")
```

と書く。最低限の記述は、これだけ。カスタマイズ用のパラメータは、元のサイトを参照。

2. 使い方

「私には emacs のカスタマイズは面倒だ。」

という文章を入力するのに、

```
watasiniha[Space][Return][Space]emacs[Space]no[Return]
kasutamaizuhamenndouda.[Space][Return]
```

とキーを打つ。ここで、emacs の入力で “s” を打った瞬間にフェンス内が、“えま c” から “emacs” に変化することと、その後で、Space キーだけで、“emacs” が確定して Space が入力されることが、egg-remix モードの特徴である。

“の” を入力した後で、[Return] で確定しているのは、その後のかな漢字変換の効率を上げるためである。変換サーバーである canna は、文章の前の方から自立語を取り出すという変換ロジック をとっているので、助詞から始まるような文章の変換は、まともにできない。使ったことはないのでわからないが、Wnn の jserver は、後方から単語を取り出して変換するらしいので、Wnn を変換サーバーに使えば、この問題は起きないのかもしれない。

“emacs” がフェンスモードでそのままアルファベットで表示されたのは、そのままでは「ローマ字」として「かな」に変換できなかったからである。ローマ字としても意味のある文字列をアルファベットに変換するためには、C-r と押す。例えば、make と入力すると、フェンス内では、“まけ” となっているので、ここで C-r を押すと、“make” に変わり、Space キーで確定すると同時にスペースを挿入する。その後で、再度、make と入力すると、今度は “make” のままになる。これは、辞書ファイル (default では ~/.remix-freq) に make をアルファベットであると登録したからである。この後で、「負け」を入力したかったら、make と入力した後で C-w を押して、“まけ” に戻してから、Space キーで変換する。C-w によって、辞書ファイルに、make はローマ字であると登録し直す。

egg-remix に固有のキーバインドをまとめておく。egg-remix は、フェンスモードでの挙動を制御しているだけなので、漢字変換モードでは、通常の tamago4/emacs のキーバインド になる。

3. アルファベットから漢字に変換 (egg-remix の変更)

表 4: egg-remix のキーバインド

[Space]	フェンスの状態がかなであれば変換モードに移行する。さもなければ入力したキー列で確定し、スペースを挿入する。
[Return]	フェンスを変換せずに確定する。
C-r	フェンスをアルファベットとして確定する。フェンスの状態がかなであった場合はユーザ辞書にアルファベットと登録する。
C-w	アルファベットと辞書に登録されているローマ字綴りをフェンス内でかなに戻す。同時に辞書にかなであると登録する。
C-c	入力をキャンセルしてフェンスを消去する。
C-t	フェンスの末尾にカーソルがある場合は最後の二文字を入れ替える。フェンスの途中にカーソルがある場合はカーソル位置とその前の文字を入れ替える。
C-k	フェンス内のカーソル以降を消去する。
ESC-h	フェンスをひらがなで確定する。
ESC-k	フェンスをカタカナで確定する。
ESC-¿	フェンス内の半角英数記号を全角にして確定する。

egg-remix モードを使ってみて不便だったのは、フェンス内がアルファベットの状態から漢字に変換ができないことだった。長らく skk を使っていたので、TU → 東北大学、ws → ワークステーションというように、読みに頭文字等を使った単語登録を利用してしたのだが、egg-remix だと、フェンス内がアルファベットの時に Space キーを押すと、確定してスペースを挿入してしまう。そこで、egg-remix.el を変更して、フェンス内が「かな」か「アルファベット」かに関わらず、変換モードに移行する関数を、C-j に割り当てた。変更した egg-remix.el は、配布パッケージに入っている。これで、アルファベットから、C-j を押すと、直接、漢字変換モードに飛ぶ。これは、便利 !! 。

6 Appendix

6.1 tgif での日本語入力

日本語 tgif は、日本語入力に kinput プロトコルと XIM プロトコルをサポートしている (その他に、中国語入力などにも対応)。どちらのプロトコルを用いるかの指定は、X リソースに記述するか、tgif のコマンドラインオプションで行なう。

一般に、X 上のアプリケーションの挙動を制御するための機構には、次のようなものがある。

1. 環境変数
   XMODIFIERS, LANG など

2. X リソースファイル
   システム全体のリソースの設定は、/usr/X11R6/lib/X11/app-defaults/ や /usr/X11R6/lib/X11/ja_JP.eucJP/app-defaults/ にある。後者は、環境変数 LANG が ja_JP.eucJP に設定されている場合のみに参照される。個人の設定ファイルは、~/.Xresources か ~/.Xdefaults ファイルである。~/.Xresources と ~/.Xdefaults の違いは、http://www.jp.freebsd.org/QandA/HTML/1032.html を参照。

3. アプリケーション独自の設定ファイル
   emacs の ~/.emacs や ~/.emacs.el ファイルのようなもの。最近のアプリケーションは、この設定ファイルをアプリケーションから直接操作できる (“options” のようなメニューで変更した内容が、設定ファイルに保存される) ものが増えている。

4. アプリケーション起動時のコマンドラインオプション

これら相互で異なった設定をした場合、一般には、下の方の設定が優先されることが多い。tgif での日本語入力を制御する方法を、表にまとめると次のようになる。

	kinput プロトコル	XIM プロトコル
X リソース	Tgif.DoubleByteInputMethod: kinput2	Tgif.DoubleByteInputMethod: xim
オプション	-dbim kinput2	-dbim xim
変換起動キー	Ctrl + Space	Shift + Space
変換終了キー	Shift + Space	Shift + Space

tgif を立ち上げて、テキスト入力モードで**日本語フォントを選んだ状態**で、Shift + Space で日本語入力にならない場合は、Ctrl + Space を試してみる。Ctrl + Space で日本語入力になる場合は、XIM プロトコルではなく、KINPUT プロトコルで、通信している。そのまま使い続けても良いが、他のアプリケーションと同様に、変換起動キーを Shift + Space にしたい場合は、tgif -dbim xim で立ち上げる。”Cannot open the X Input Method” と言われる場合は、環境変数 XMODIFIERS が設定されていないか、kinput2 が立ち上がっていないので、チェックする。

毎回、tgif -dbim xim として立ち上げるのは面倒なので、~/.cshrc ファイルで、

```
alias tgif tgif -dbim xim
```

と書いておけばよい。

同様な設定を、X リソースで行うためには、次のようにする。個人の X リソースの設定ファイルは、~/.Xdefaults か ~/.Xresources である。.Xdefaults と .Xresources の違いは、こちらを参照。どちらかのファイルに、

```
Tgif*DoubleByteInputMethod: xim
```

と書けば、次回の login から反映されるはずである。ちなみに、現在の X リソースの設定は、

```
% xrdb -query
```

で見ることが出来る。

6.2 canna に付属する utulity プログラム

詳しい使用法や コマンドラインオプション は、

```
% man XXXXX (英語)
% jman XXXXX (日本語)
```

のマニュアルページを参照。

1. 変換サーバーのチェック

   cannacheck 「かんな」の関連情報の表示

   cannastat かな漢字変換サーバの情報の表示

   cshost かな漢字変換システムのサーバ・アクセス制御

2. cannaserver と通信して、辞書を管理するツール

   catdic 辞書の内容を表示する

   cpdic ユーザ辞書コピーツール

   chmoddic 辞書のアクセス権を変更する

   addwords 一括単語登録

   delwords 一括単語削除

   lsdic 辞書の一覧の表示

3. ローカル辞書のメンテナンスツール

   dicar バイナリ辞書のアーカイブ

   dpbindic バイナリ形式の辞書ファイルの単語情報を出力

mkbindic テキスト形式の辞書からバイナリ形式辞書と頻度ファイルを作成

mkdic 辞書の作成／アップロードを行う

mvdic ユーザ辞書名前変更ツール

rmdic ユーザ辞書の削除ツール

splitword 辞書ファイルを１行１候補にする

4. Wnn の辞書との互換ツール

ctow Wnn のテキスト形式の辞書への変換

wtoc Wnn のテキスト形式の辞書からの変換

5. ローマ字仮名変換テーブルの操作

dpromdic バイナリ形式のローマ字かな変換テーブルのテキスト形式への変換

mkromdic ローマ字かな変換テーブルを作成する

6.3 emacs lisp のバイトコンパイル

例えば、~/.emacs や ~/.emacs.el に、

```
(load "egg-remix")
```

という行があったとき、emacs は、load-path の directory で、egg-remix.elc 、egg-remix.el というファイルを順番に探し、あれば読み込み、なければエラーを返す。この、egg-remix.elc は、emacs lisp のソースファイル (egg-remix.el) を コンパイル (emacs 用語では、バイトコンパイルと呼ぶ) して出来たファイルである。バイトコンパイルすると、読み込みが高速になるので、emacs の起動が速くなる。しかし、ソースファイルである .el を変更しても、.elc が存在すれば、そちらが読み込まれるので、一度、.elc を作成したら、.el を変更する度に、バイトコンパイルする必要がでてくる。したがって、頻繁に変更するファイル (~.emacs.el など) については、バイトコンパイルせずに、ほとんど変更しないファイルについては、バイトコンパイルしておいた方がよい。

バイトコンパイルの方法は、emacs を立ち上げた後、

```
ESC-x byte-compile-file
```

でおこなうか、または、コマンドラインで、

```
% emacs -batch -f batch-byte-compile *.el
```

でおこなう。

6.4 参考 URL 一覧

- 日本語入力プログラムについて考える
  http://www4.airnet.ne.jp/koabe/com_inet/im/
- 日本語入力システム「かんな」(現在の開発元)
  http://canna.sourceforge.jp/
- 日本語入力システム「かんな」(NEC)
  http://www.nec.co.jp/japanese/product/computer/soft/canna/index.html
- canna に関するリンク集
  http://canna.sourceforge.jp/links.html
- えせかんな
  変換クライアントには cannaserver を装い、実際には、ATOK や Wnn6 などの変換サーバーと通信する
  http://esecanna.netfort.gr.jp/
- .Xdefaults と .Xresources の違い
  http://www.jp.freebsd.org/QandA/HTML/1032.html
- X リソースについて (X の日本語マニュアルより)
  http://xjman.dsl.gr.jp/man/man3/X.3x.html#lbAO
- tamago4 のローマ字-かな変換のカスタマイズ
  http://www.ht.sakura.ne.jp/~delmonta/emacs/25.html
- tamago4 のカスタマイズ
  http://sodan.org/~knagano/emacs/tamagov4.html
  http://phe.phyas.aichi-edu.ac.jp/~cyamauch/pl2_emacs21.html
- canna 辞書を強化するプロジェクト
  http://cannadic.oucrc.org/
- canna の変換ロジック
  http://www.jaist.ac.jp/~fujieda/canna/Canna-2257
- Wnn の jserver の変換ロジック
  http://www4.airnet.ne.jp/koabe/com_inet/im/feature.html#wnn
- 検索で日本語入力できない tamago4 のバグ
  http://phe.phyas.aichi-edu.ac.jp/~cyamauch/pl2_emacs21.html
- tamago4 でも canna の機能を使う
  http://cgi18.plala.or.jp/nyy/canna/

- アルファベット入力から直接漢字に変換 (boiled-egg)
  http://usir.kobe-c.ac.jp/boiled-egg/
- アルファベット入力から直接漢字に変換 (boiled-canna)
  http://www.ceres.dti.ne.jp/~knak/boil.html
- アルファベット入力から直接漢字に変換 (boiled-egg on tamago4)
  http://www.gcd.org/sengoku/boiling-egg/
- canna/mule で 日本語/アルファベットを効率良く入力 (egg-mix)
  http://www.maa.sst.ne.jp/ nsmrtks/egg-mix/
- tamago4 で 日本語/アルファベットを効率良く入力 (egg-remix)
  http://www.extipl.jp/~payashi/remix/)