Vol.2013-MBL-65 No.3
Vol.2013-UBI-37 No.3
2013/3/14

スマートフォン向けテキスト入力システム

鈴木 孝宏[1,a)]　美馬 義亮[1]

概要：スマートフォンのテキスト入力は，ソフトウェアキーボードとよばれる物理的なキーボードを模したもので実現されている．しかしながらこのインタフェースは，もとはと言えばスマートフォンの使用環境とは異なる，オフィス内利用のインタフェースである．それによって，ボタンが密集するなどの理由から入力がしづらいという問題を抱えている．本研究では，このスマートフォンの入力のタップ数に着目し，新たなテキスト入力システムの提案を行う．単語単位の入力でタップ数を減らすことができるシステム「LiteText」を実装し，テキスト入力にかかる所要時間やタップ数をソフトウェアキーボードと比較して調査した．結果，所要時間はソフトウェアキーボードの方が早かったが，タップ数は LiteText の方が少なかった．また，被験者へのインタビューから，タップ数を減らすことが入力のしやすさに繋がることが分かった．

1. はじめに

1.1 本研究の着目点

スマートフォンはタッチパネルでシステムを操作し，電話やメール作成をする．近年では SNS サイトにつぶやきを投稿したりすることもできるようになった．こうした機能を利用するにあたって，テキスト入力は欠かすことのできない重要な機能である．本研究では，スマートフォンのテキスト入力に着目し，新たなテキスト入力方式の提案を行う．

1.2 テキスト入力の現状

まずは入力に使用するインタフェースについて見てみる．スマートフォンのテキスト入力は，ソフトウェアキーボードと呼ばれる，画面上にキーボードをシミュレートしたソフトウェアによって実現されている．入力するテキストの種類に応じて，キーボードを切り替えることができ，QWERTY キーボード（**図 1**），テンキー，テンキーにかなの行を割り当てた携帯かな配列（**図 2**）などがある[*1*2]．基本的な入力方法は，入力したいキーに触れることで，テキストを入力することができる．また携帯かな配列には，キーに置いた指を上下左右にはじいて入力する「フリック入力」という入力方法も選択できるようになっている．

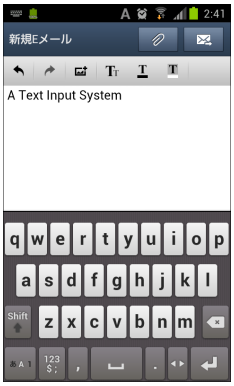

図 1　QWERTY 配列
Fig. 1　QWERTY Keyboard

図 2　携帯かな配列
Fig. 2　Numeric keypad for Kana Input

次に，スマートフォンにおけるテキスト入力の環境や内容について見てみる．スマートフォンのテキスト入力環境は，例えば電車やバスの中，場合によっては歩きながらでも行われ，不安定な環境であることが多い．また，テキストの内容は「これから帰ります」「ゼミに遅れます」といったような短めのものになっている．これらについてパーソナルコンピュータでは，環境はオフィス内利用の安定した場所であり，入力されるテキストもじっくり吟味しながら入力する長文である．スマートフォンとは大きく異なっていることが分かる．

1.3 問題点

QWERTY キーボードやテンキーは，小さなスマート

[1] 公立はこだて未来大学 システム情報科学部 情報アーキテクチャ学科
[a)] b1009154@fun.ac.jp

[*1] 図は全て，Android 用アプリケーション「Google 日本語入力」[1] のスクリーンショットである．
[*2] 本論文における「携帯かな配列」は，Google 日本語入力では「ケータイ配列」という名称になっている

フォンの画面に表示するにはキーの数が多い．その結果，キー同士の間隔が狭く操作しづらいという問題点を抱えており，急ぎの連絡をする際は，円滑な入力ができずに困ることがある．このことから，キーの数を極力減らした，シンプルな入力システムが必要であると言える．

また，フリックで入力する携帯かな配列だが，はじいたり，画面から指を何度も離すことは，不安定な状況では入力が難しい．こうした状況でも安定した入力を実現するには，何度も指を離さなずにテキストを入力できるシステムが必要である．

1.4 研究目標と提案

そこで，不安定な状況でも確実に入力できるスマートフォン向けテキスト入力システムを提案する．小さな画面でも操作しやすいシンプルなインタフェースで設計し，入力にかかるタップ数を減らすことを目標とする．具体的には，1 文字ずつのテキスト入力ではなく単語単位で入力できるようにすることが，システムの重要な条件となる．

実装の方針としては，画面に表示した 3 行 3 列のフィールド上で，指を上下左右に移動させることでテキストを選択する方法にした．入力できるテキストは予め限定した．各テキストにパラメータを設定しておき，入力時には指の動きによって生成したパラメータをキーに，テキストを選択できるようにした．

2. 関連研究・関連事例

2.1 なめらかなユーザインタフェース

直感的なインタフェースの捉え方として，なめらかなユーザインタフェースという概念が提唱されている [2]．これは，ユーザが操作しやすいインタフェースの性質を考えたもので，連続性・可逆性・直接性・直感性の 4 つの性質を持っているものである．本研究で提案するシステムでは，これを参考に，連続性・可逆性を持たせた設計にした．

2.2 Dasher

Dasher[3], [4] は，ポインティングデバイスのみでテキスト入力を可能にしたソフトウェアである．ユーザは，画面に指を置き，流れてくる文字から入力したいものを選ぶことで文章を作成する（**図 3**）[*3]，入力中は画面に指を置いたままであるため，1 つの文章を入力するために要するタップ数は 1 回と言って良い．しかしながら，入力にかかる時間は，文字が流れてくるスピードや目的の文字への距離によって大きく変わる．

Dasher はポインティングデバイスのみで入力を実現しており本研究と共通する部分はあるものの，1 文字単位の入力であり，本研究で提案するような単語単位の入力とは異なるものである．

[*3] Android 用アプリケーション「Dasher」のスクリーンショットである．

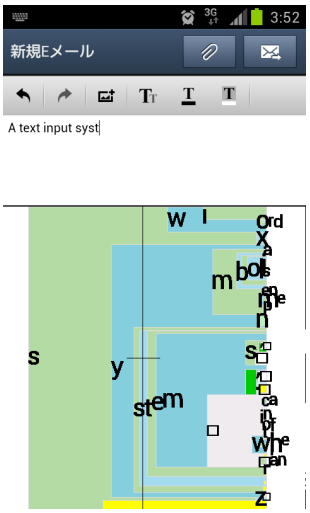

図 3 Dasher の入力画面
Fig. 3 Input Screen of "Dasher"

図 4 LiteText のスクリーンショット
Fig. 4 Screenshot of "LiteText"

2.3 携帯電話の予測変換

POBox[5] に代表される携帯電話の予測変換は，ユーザが入力したテキストに基づいて入力の途中でも変換候補を表示したり，過去の入力やユーザの癖に基づいて入力候補を表示したりするシステムである．こうした入力予測・予測変換は，携帯端末には欠かせない機能になっており，テキストを入力する際のタップ数を減らす手助けになっている．本研究では，こうした予測変換ではないものの，ユーザの入力に基づいて，次の操作でどのようなテキストを入力できるかを提示する機能を持たせることにした．

3. 提案システム「LiteText」

この章では，本研究で開発した「LiteText」について述べる．まず，LiteText のスクリーンショットを**図 4** に示す．

LiteText は，入力フィールド上を指がどのように動いた

図 5 「読む」の入力方法
Fig. 5 How to input “読む”

かをパラメータに，単語単位でテキストを生成するアプリケーションである．これによりユーザは，複数文字で構成されるテキストを，1 度のタップで入力できるようになる．以降，LiteText の詳細を述べる．

3.1 LiteText のインタフェース

LiteText は，大きく 2 つの部分に分けることができる．9 分割入力フィールド，5W1H 切替ボタンの 2 部分に分かれている．

3.1.1 9 分割入力フィールド

LiteText は，3 行 3 列で分割された 9 分割入力フィールドを，指でくるくるとなぞり，移動しながらテキストを入力する．フィールドには，入力することのできるテキストを表示している．入力したいテキストが表示されているフィールドに指を置くことでテキストを入力できる．入力したいテキストが表示されていない場合は，それに関連しそうなテキストのフィールドに指を置く．すると，そのフィールドの上下左右には関連テキストが表示される．指を画面から離さず，入力したいテキスト（あるいは，最終的に入力したいテキストに関連しそうなテキスト）のフィールドにスライドすることで，入力するテキストを変化させることができる．例として，「読む」と入力する場合の指の動かし方を**図 5** に示す．

この入力方法は，一見するとフリック入力にも見えるかもしれない．しかしながら，指の動かし方と入力されるテキストに違いがある．フリック入力は，キーを選択するために指を一方向に一度だけ動かし，ひらがな一文字を入力する．一方，9 分割入力フィールドでは，指を上下左右に移動することで辞書の木構造の深部まで進み，テキストを入力する．言い換えれば，9 分割入力フィールドには，テキストにたどり着くための道順があるということである．この点がフリックと大きく異なる部分である．

3.1.2 5W1H 切替ボタン

LiteText は 5W1H モデルを参考にテキストを生成できるように設計している．5W1H は，文章を記述する際に必要な要素を表したモデルである．具体的には，いつ・どこで・だれが・なにを・なぜ・どのように，といった 6 つの要素で記述する考え方である．これを参考にし，LiteText では，いつ・どこで・だれが・なにを・どうした（動詞）の 5 要素（分類）の単語を入力できるようにシステムを設計した．こうすることにより，テキストを記述する際のコンテクストを設定し，テキスト選択の絞り込みを行いやすくするメリットがある．実際の入力の際には，入力したいテキストが割り当てられているであろうカテゴリを 5W1H 切替ボタンから選択し，9 分割入力フィールドを指でなぞり，テキストを入力するという流れになる．

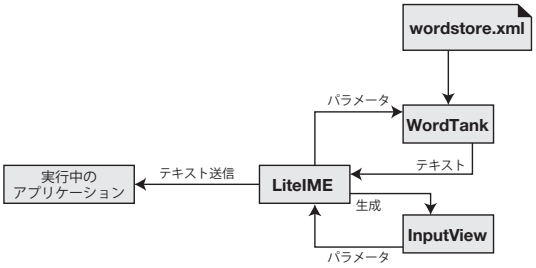

図 6 LiteText を構成するクラスの関係図
Fig. 6 Class Diagram of LiteText

3.2 LiteText の実装

ここではまず，LiteText の実装を，クラスの関係図で示す（**図 6**）．

3.3 文字列生成エンジン

LiteText が文字列を生成するプロセスは，4 ステップになる．(1) ユーザが選択している 5W1H 切替ボタンと 9 分割入力フィールドへの入力から，InputView クラスが後述のパラメータを生成，(2) そのパラメータを LiteIME クラスに渡す，(3)LiteIME クラスは WordTank クラスにパラメータを送るのと引き換えに，それに対応したテキストを取得する，(4)LiteIME クラスはそれをアプリケーションへと送信している．

3.3.1 パラメータの生成

では，ユーザが入力したパラメータから，どのようにしてテキストを生成しているかを述べる．InputView クラスは，現在選択されている 5W1H 切替ボタンに割り当てられたカテゴリコード（A から E の英字）と，ユーザが通ったフィールドを順に記録した文字列（0 から 8 の整数）を並べることでパラメータを生成している．割り当ての詳細は，**図 7** に記した．たとえば，図 5 の「読む」の場合，5W1H 切替ボタンで「どうした」(E) を選択し，入力フィールドを 4 → 5 → 2 の順に指を動かす．こうして「E452」という文字列がパラメータになる．なお，ユーザが 4 → 5 → 2 → 5 と戻った場合，パラメータは「E45」になる．

3.3.2 パラメータとテキストの関連づけ

LiteText は，入力できるテキストの辞書を予め XML で

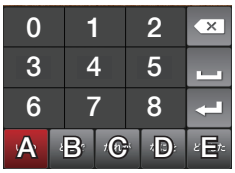

図 7 各フィールドの番号と，5W1H 切替ボタンのコード
Fig. 7 Numbers of each Input Field and Codes of 5W1H change switch

用意した．WordTank クラスが XML を読み込み，パラメータとテキストをハッシュマップに格納している．

3.4 表現できるテキスト

LiteText は，主として卒業研究を行う学生が使用することを想定し「研究室」「卒業研究」「発表」などのテキストを辞書に用意した．また，使用する状況をバスや電車内で急な連絡を行う，SNS サイトに自らの置かれている状況をつぶやくといった状況を想定し，「バス」「遅れる」「課題」「なう」なども用意した．文章を作る上で必要になる助詞だが，今回の実装では扱わないこととした．表現できる作文例としては，「明日，10 時 40 分，私，卒業研究，発表．大学．」といったようなものになる．

4. 評価実験と結果

従来のソフトウェアキーボードには，不安定な状況におけるテキスト入力が困難であった．実験では，歩きながら入力することで不安定な状況を作り，LiteText の入力にかかる所要時間と，画面をタップした回数を比較した．また，ユーザビリティや印象に関する調査も行った．

4.1 実験方法

被験者は，公立はこだて未来大学の学生 6 人（スマートフォン所有者）であった．

4.1.1 使用した道具

実験に使用した道具は，スマートフォン（LiteText と Google 日本語入力をインストールしたもの），時計（スマートフォンに時刻を同期させてある），パーソナルコンピュータ（操作ログの記録のため）であった．

4.1.2 実験の手順

実験は，テキストを入力する課題を与え，通常のソフトウェアキーボードを使った場合と本システム LiteText を使った場合の 2 つの条件で行った．ここではまず，ソフトウェアキーボードを使用した場合について述べる．

実験前の準備 実験で使用するソフトウェアキーボード「Google 日本語入力」の入力方法を被験者が普段入力する入力方法に設定しておいた．また，学習機能は解除しておいた．また，予めメモ帳アプリケーションを起動しておいた．

テキスト入力タスク まず携帯かな配列のソフトウェアキーボード（Google 日本語入力）の操作に慣れてもらう時間を 5 分設け，実行しておいたメモ帳アプリケーションにテキストを自由に入力してもらった．被験者は安定した椅子に座った状態であった．5 分後，被験者が希望した場合は，さらに練習の時間を設けることにした．

その後，指定したテキストを入力する課題を与えた．被験者には歩きながら入力してもらうことで．不安定な入力状況を作ることにした．被験者は入力し始めると同時に歩き出してもらい，入力を終えたら足を止めてもらった．歩き出した時刻と，足を止めた時刻を記録した（操作ログを抽出する際の目安に使用した）．

一通り入力を終えたところで，スマートフォンとパーソナルコンピュータと接続し，記録しておいた時刻をもとに操作ログを抽出した．

この手順を，LiteText でも同様に行った．LiteText の場合は，実験を始める前に，被験者に対して LiteText の操作方法を予め用意した説明書で説明した後，5 分間の練習時間を設けた．

2 条件のテキスト入力タスクの終了後，被験者にはアンケートに答えてもらった．操作方法の覚えやすさ，入力の仕方，使用してみた感想を調査した．

入力するテキストパターン 入力するテキストは，パターン 1「おはようございます。大学なう。」，パターン 2「10 時 40 分、授業欠席。」，パターン 3「明日、卒業研究、発表。」，パターン 4「友達、本読む。私、課題なう。」，パターン 5「家帰る。バス遅れる。」の 5 つを用意した．

4.2 結果

4.2.1 タップ数

テキストパターンごとの入力に要したタップ数を**図 8** に示す．タップ数は，ソフトウェアキーボードよりも LiteText の方が少なかった．この 2 条件の差が最も大きかったのはパターン 2「10 時 40 分、授業欠席。」だった．パターン 2 の入力に要する平均タップ数は，ソフトウェアキーボードでは 85.17 回，LiteText は 11.17 回であった．パターン 4 では，ソフトウェアキーボードが 101.67 回，LiteText が 34.33 回とおよそ 3 分の 1 のタップ数だった．

4.2.2 所要時間

テキストパターンごとの入力の所要時間を**図 9** に示す．全体的に，ソフトウェアキーボード，LiteText 双方ともほぼ同じか，LiteText の方が入力に時間がかかった．タップ数の差が最も大きかったパターン 2 の平均所要時間は，ソフトウェアキーボーでは 28.25 秒，LiteText では 26.97 秒と，LiteText の方がわずかに短かった．

4.2.3 印象評価: 操作方法は覚えやすかったか

操作方法は覚えやすかったかの回答の結果を**図 10** に示す．8 割を超える被験者が，あてはまる，またはやや当てはまると答えた．

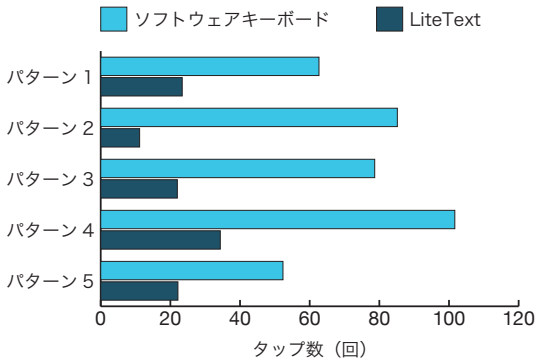

図 8 テキスト入力に要した画面のタップ回数
Fig. 8 A Number of Taps for Text Input

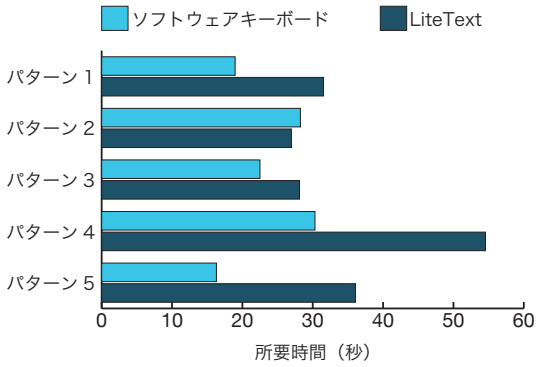

図 9 テキスト入力にかかった所要時間
Fig. 9 Amount of Time Required for Text Input

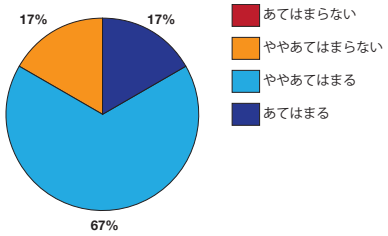

図 10 「操作方法を覚えやすかったか」の回答
Fig. 10 Answers of “Learning use of LiteText was easy”

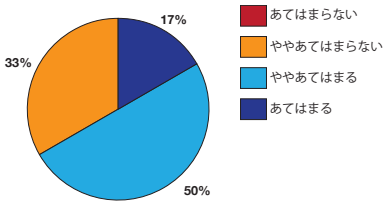

図 11 「歩きながらでも入力しやすかったか」の回答
Fig. 11 Answers of “Walking Input was easy”

4.3 印象評価: 歩きながらでも入力しやすかったか

歩きながらでも入力しやすかったかの回答の結果を**図 11** に示す．7 割の被験者は当てはまる，やや当てはまると答えた．一方で，3 割はやや当てはまらないと答えた．当てはまらないと答えた被験者はいなかった．

4.3.1 自由記述回答

自由記述部分は，LiteText を使用した感想，高く評価できる点，要望や改善のアドバイスを自由に書いてもらった．

まず，LiteText を使用した感想では，以下の感想を得ることができた．

- 指を手前にスライドしたいときに，手前のテキストが指に隠れて見えづらい
- カテゴリが難しい，テキストを探すのが大変
- ツリー構造がしっかりしていると分かりやすい
- 慣れが必要だと感じた

一方で，

- 慣れればフリック入力だけでなく LiteText も活用したい
- 探すのに時間はかかるが覚えやすい
- 定型文の入力にはぴったし

といった，ポジティブな感想も貰うことができた．

高く評価できる点は，入力の方法を評価するものが目立った．内容としては，

- タップ数が減った
- フリック入力のように指をスライドすることで入力できる
- 「。」「、」の入力の仕方が直感的で良い

といったものであった．

要望や改善のアドバイスとしては，入力できる単語のカスタマイズや追加や，システムからのフィードバックに関するものがあった．

- 助詞，アルファベット，時間は 1 分単位で入力したい
- 単語の登録をしたい，単語を増やしてほしい
- 入力履歴からの学習機能や，ライフログからのカスタマイズができるといい
- 予測変換とうまく組み合わせるといい
- 押しているフィールドで入力できる単語のフィードバックを見やすくする

5. 考察

5.1 タップ数を減らし，安定した入力を実現

タップ数は，ソフトウェアキーボードを用いた場合と比較して，全ての試行でタップ数は減少している．この一番の要因として，LiteText が 1 度のタップでまとめてテキストを入力するという点が挙げられる．

また，歩きながら入力する場合，ソフトウェアキーボードと比べて LiteText の方が入力しやすかったという結果が出ている点を見ると，タップ数が少ない方が入力しやすいということが分かる．ソフトウェアキーボードの場合，1 文字入力するごとに指が画面から離れ，次に入力するキーの位置を確認してから入力する必要がある．一方 LiteText の場合は，ソフトウェアキーボードのように画面から指が何度も離れることはない．これが，入力しやすいことの要因であり，安定した入力を実現できたポイントであると考えられる．

5.2 LiteText の方が入力に時間を要した点について

ソフトウェアキーボードと比較して，LiteText のテキスト入力にかかる時間が，遅くなったかほぼ同じだった．LiteText で入力に時間がかかった原因として，入力したいテキストを探すために時間がかかったことが挙げられる．自由記述回答でも「テキストを探すのが大変」と答えた被験者がいたことからもそれが伺える．また「テキストのツリー構造を分かりやすくしてほしい」という被験者もいた．辞書の構造をもっと分かりやすくする必要があり，辞書の設計にまだ工夫の余地があると考えられる．

また「慣れが必要」という回答にもあるように，指の動作と入力されるテキストを結びつけるだけの十分な練習時間を設けていなかったことも原因と考えられる．一方で，1 回の練習時間ではなく使用した回数で慣れることも考えられるため，今後は，同じ被験者で何度か実験を行うことも視野に入れる．

5.3 ライフログなどを活用した展開の可能性

自由記述の回答にもあったように，ライフログを活用してカスタマイズできるようにしてはどうかといった意見が出た．例えば，GPS を活用して，ユーザが過去に訪れた場所，あるいは現在地の場所を入力できるようにしたり，近所の施設名をリストアップするなどの可能性が挙げられる．他にも，訪問した Web サイトや，メールやチャットで話題に上がっているキーワードを抽出することで，文脈に沿ったテキストを入力しやすくする可能性も挙げられる．

一方で，システムの挙動が常にチューニングされる場合，ユーザが操作を覚えるのが難しくなる可能性がある．ライフログを活用したシステムにする場合は，この部分が使い勝手の鍵になると考えられる．

6. おわりに

スマートフォンのテキスト入力は，ソフトウェアキーボードを画面に表示することで実現されている．しかしながら，スマートフォンの画面の小ささのために操作しづらかったり，画面から指が何度も離れたりすることによって，電車やバス内，歩きながらといった不安定な状況での円滑な入力が難しいという問題があった．本研究では，入力にかかるタップ数に着目し，操作しやすいテキスト入力システム「LiteText」を提案し，評価実験を行った．

6.1 テキスト入力のタップ数を減らすことができた

テキスト入力にかかる所要時間はソフトウェアキーボードよりも時間がかかったものの，画面をタップする回数が減少した．この「タップ数を減らす」ことは，入力のしやすさにも繋がるということが，被験者へのアンケートの結果から分かった．

6.2 見えてきた問題点

シンプルなインタフェースであるといった感想を得ることができた一方で，入力されるテキストを探す手間や，テキスト修正にはバックスペースキーを何度も押さねばならないという問題点も見えてきた．

6.3 今後の展開

システム面では，ユーザ辞書に対応することで，よりユーザが使いやすい実用的なシステムにできると考えられる．また，学習機能を付けてユーザの入力傾向でチューニングしたり，ライフログや GPS センサ等も活用することで，よりコンテクストベースのテキスト入力システムを実現することも考えられる．これは，従来ユーザからの指示を待っていた入力システムが，ユーザ側に一歩近づくことになり，利便性の向上が期待できる．

参考文献

[1] Google： Google 日本語入力 (online), 入手先 <http://www.google.co.jp/ime/> (2013-01-23).
[2] 増井 俊之, 水口 充, George Borden, 柏木 宏一: なめらかなユーザインタフェース, 第 37 回冬のプログラミングシンポジウム予稿集, pp. 13-23. 情報処理学会 (1996).
[3] Inference Group: Dasher Project(online), 入手先 <http://www.inference.phy.cam.ac.uk/dasher/> (2013-01-23).
[4] Wikipedia: Dasher(online), 入手先 <http://ja.wikipedia.org/wiki/Dasher> (2013-01-23).
[5] 増井 俊之: POBox(online), available from <http://www.pitecan.com/Index/pobox.html>(2013-02-10).