分散入力メソッド IIIMF における IIIMF-SKK の実装

市村元信
Momonga Project
早稲田大学

2002 年 9 月 18 日

概要

分散入力メソッド IIIMF(Internet/Intranet Input Method Framework) は、Language Engine と呼ばれるモジュールを用意する事で、各言語に対応した Input Method(以下 LE) を提供する事ができる Framework である。本稿では、LE の実装の一つである IIIMF-SKK について解説する。また、今後の展開、他の LE への応用についても述べる。

1 はじめに

IIIMF(Internet/Intranet Input Mothod Framework) は、従来広く利用されてきた XIM(X Input Method) では対応しきれなくなった問題を解決するために考案された Input Method である。IIIMF を利用する事で、OS 非依存で、X Window System にも依存しない形で入力メソッドを提供する事が可能となった。IIIMF を利用して入力メソッドを提供するためには、Language Engine Interface Specification[1] で規定されている LEIF(Language Engine Interface) を実装したモジュールを作成する。Linux 上で動作する IIIMF の実装としては、Linux の標準化団体の一つである LI18NUX[*1]において提供されている im-sdk が存在する。im-sdk は、入力メソッドを開発するための SDK としても利用する事が可能である。著者は、この im-sdk を利用し、LEIF を実装した LE である IIIMF-SKK を開発、公開している。本論文では、まず、IIIMF の概要を述べ、次に LEI を実装した LE である IIIMF-SKK の内部について説明する。また、今度の IIIMF の動向を含め、IIIMF-SKK の今後、他の LE への応用等についても述べる。

[*1] http://www.li18nux.org

2 IIIMF

IIIMF は、JavaOS への対応を始めとして、X Window System に依存しない形での入力メソッドを提供する事をきっかけとして策定され、Solaris 上で利用されてきた。2000 年に Sun Microsystems はこの IIIMF をオープンソース・ソフトウェアとして公開した。IIIMF は現在 Linux にも移植されており、LI18NUX においてメンテナンスが行われている。IIIMF は、従来の入力メソッドが単一のユーザに対してサービスを行っていたのに対し、Web サーバの様に、ネットワーク上に入力メソッドを提供するサーバとして動作する事で、複数のユーザに対して入力メソッドを提供する事ができる。また、IIIMF は、従来の入力メソッドが単一の言語、単一の入力方式を提供するのに対して、複数の言語、複数の入力方式を提供する事ができる。本節では、Linux 上で動作する IIIMF の実装の一つである im-sdk について、入力メソッドとしての特徴、動作概要、LE の実装に必要な事項について述べる。

2.1 IIIMF の特徴

IIIMF の特徴を図 1 に示す。IIIMF は、前述したように、複数の言語、複数の入力方式を提供す

る Framework であり、内部の文字コードとしては、UTF-16 を利用する。IIIMF の実装である im-sdk では、IIIMF 内部で利用する UTF-16 と従来の文字コードとの変換には、IBM が公開している ICU(International Components for Unicode)[*2]を利用している。IIIM クライアントと IIIM サーバ間の通信は、IIIMP(Internet/Intranet Input Method Protocol) を利用して行う。

- Multiplatform, platform independent
- Multilingual/Full UNICODE support
- Windowing System Independent
- Multiple language engines concurrently run
- Multiuser
- Distributed, lightweight client and server
- Extensible
- Efficient input method protocol

図 1: IIIMF の特徴

2.2 IIIMF の動作

IIIMF は、図 2 に示すように、IIIMF クライアント用の Framework である IIIMCF(Internet/Intracet Input Method Client Framework) と IIIM サーバ用の Framework である IIIMSF(Internet/Intranet Input Medthod Server Framework) とからなり、これらの各 Framework が IIIMP(Internet/Intranet Input Method Protocol) を利用して通信する事で構成される。利用する事が可能な LE は、図 3 で示すように IIIMSF から IIIMCF に送信される。現在 LI18NUX で公開されている im-sdk では、IIIMCF の実装として

- IIIMECF(Internet/Intranet Input Method Emacs Client Framework)
- IIIMJCF(Internet/Intranet Input Method Java Client Framework)
- IIIMXCF(Internet/Intranet Input Method X Client Framework)

が入手可能であり、IIIMSF の実装としては htt_server が利用できる。ここでは、特に IIIMXCF の実装である xiiimp.so を利用した入力環境について述べる。

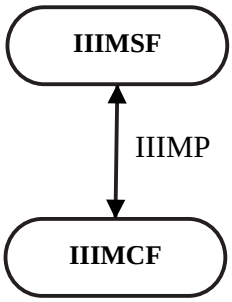

図 2: IIIMF の動作

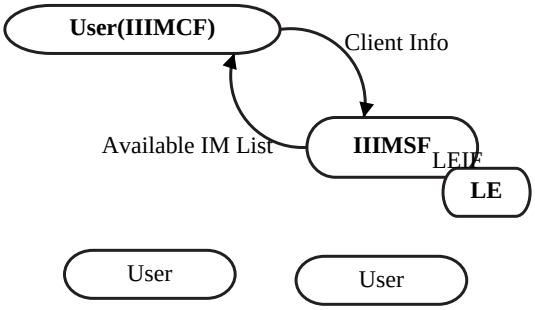

図 3: 利用可能な LE の問い合わせ

2.2.1 XIM としての利用

IIIMF の実装である im-sdk には、htt_xbe が同梱されている。htt_xbe は XIM サーバとして動作する。図 4 に示すように、htt_xbe は IIIMSF の実装である htt_server と XIM クライアントの通信を橋渡しする。これにより、XIM に対応したアプリケーションは、カスタマイズする事なく IIIMF を利用する事が可能となる。

2.2.2 Xlib-I18N からの利用

LI18NUX において提供されている Xlib-I18N を利用する事で、IIIMXCF の実装である xiiimp.so を直接 XIM オブジェクトとして扱う事が可能となる。前述したように htt_xbe を利用した場合、X のクラ

[*2] http://oss.software.ibm.com/ icu

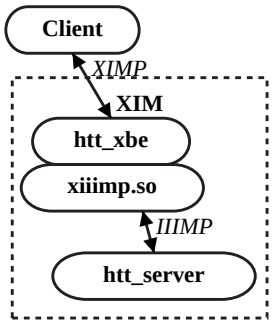

図 4: XIM として利用

図 5: htt_xbe を利用した xterm

イアントアプリケーションは、通常の XIM サーバと同じ制約を受ける事になる。一方で、Xlib-I18N を利用する事によって X のクライアントアプリケーション側で、XIM オジェクトとして IIIMXCF を利用する事が可能となる。これにより、XIM サーバの制約を受ける事なく IIIMF を利用する事が可能となる。

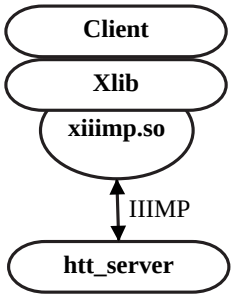

図 6: Xlib-I18N から利用

2.3 LEIF

IIIM サーバは、LEIF で規定された関数を実装した LE をダイナミックロードする事によって IIIM クライアントに対して入力メソッドを提供する事が可能となる。LE を実装する際に必要な LEIF を図

図 7: Xlib-I18N を利用した xterm

8 に示す。ここで示した各関数は、対応したイベントが生じた際に呼びだされ、LE は各々のイベントに対応した処理を行う事が可能になる。

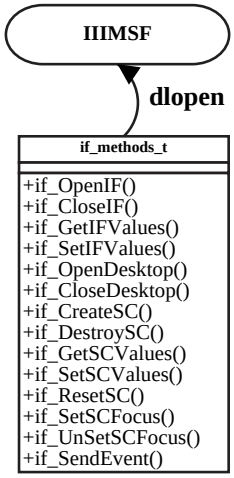

図 8: LEIF

2.4 X Auxiliary Object

IIIMF では、通常ネットワークを利用して入力の制御を行うが、パレット等の補助的な手段を提供するための機構が存在する。X Window System 上では、X のイベントのメカニズムを利用した手段として、X Auxiliary Object が利用できる。X Auxiliary Object は、図 9 に示すように IIIMXCF とデータを送受信する事で制御する事が可能になる。

3 IIIMF-SKK

IIIMF-SKK は、SKK OpenLab[*3]で開発されている SKK(Simple Kana Kanji conversion program, an input method of Japanese) と似た入力方式を提供す

[*3] http://openlab.ring.gr.jp/skk

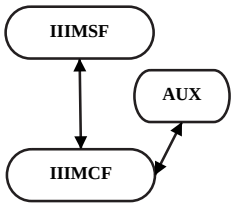

図 9: AUX

る LE であり、現在、SourceForge.jp [*4]上において開発を行っている。IIIMF-SKK は、大きく分けて

- LEIF(skk.so)
- LibSKK(libskk.so)
- X Auxiliary Object

の 3 つのコンポーネントからなり、

- glib (1.2.x or 2.0.x)
- gtk+ (1.2.x or 2.0.x)
- Berkeley DB (3.0 or higher)
- libxml2

を利用して実装を行った。

3.1 skk.so

skk.so は、IIIMF-SKK の本体であり、後述する LibSKK を利用して図 8 で示した LEIF を実装している。また、X Auxiliary Object との通信も行なっている。

3.2 LibSKK

LibSKK は、IIIMF-SKK を作成する際に、SKK の動作をエミュレートする事を目的として作成した。できるだけ再利用する事を可能にするため、IIIMF に依存しない形で実装を行った。実装に際しては、gtk+を参考にしており、

- オブジェクト指向
- イベント駆動

で、簡潔な記述が可能なように実装した。LibSKK の利用例を図 10 に示す。また

- XML をベースとした設定ファイル

[*4] http://sourceforge.jp

```
+ SkkFunc
    + SkkFuncItem
    + SkkFuncResult
+ SkkQuery
    + SkkQueryItem
        + SkkQueryFunctionImpl
+ SkkConf
    + SkkConfItem
+ SkkBuffer
    + SkkMode
    + SkkCursor
```

図 12: LibSKK の階層

- サーバへの問い合わせ部分をモジュール化

する事により、拡張しやすいように設計した。LibSKK で用いられる設定ファイル用の DTD を図 11 に示す。

3.2.1 階層図

LibSKK で提供されるオブジェクトの (概念的) な階層図を図 12 に示す。また、各オブジェクトの役割を表 1 に示す。

SkkFunc	キーボードからの入力の制御
SkkQuery	辞書への問い合わせ
SkkConf	設定ファイルの読み込み
SkkBuffer	文字バッファの制御
SkkMode	モードの制御
SkkCursor	カーソルの制御

表 1: 各オブジェクトの役割

3.2.2 Query モジュール

LibSKK では、辞書サーバ等への問い合わせはプラグインとして実装した。プラグインが実装すべき Interface を図 13 に示す。問い合わせ部分をプラグイン化した事によって、多種多様な形式の問い合わせを実装する事が可能となった。現在実装されているプラグインを表 2 に示す。

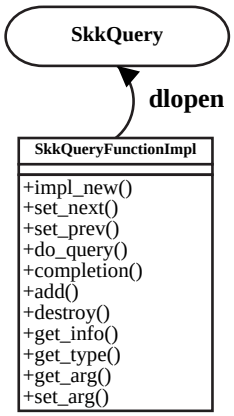

図 13: SkkQueryFunctionImpl

skkldictimpl.so	ローカル辞書
skkdictimpl.so	skkserv への問い合わせ
skklookimpl.so	look コマンドを利用した英単語補完
skkpoboximpl.so	pbserver への問い合わせ
skkcannadictimpl.so	cannaserver への問い合わせ

表 2: 実装済みの SkkQuery プラグイン

3.3 X Auxiliary Object

IIIMF-SKK では、X Auxiliary Object として、

- PaletteAux
- AddDictAux

を実装した。これらの Object は、SKK のモードを管理するといった部分を除いて、LibSKK、skk.so から独立した形で実装した。これにより、他の LE を開発する際に再利用できる事を可能にした。図 14、 15、 16 にそれぞれの図を示す。PaletteAux はパレット部分を担当する X Auxiliary Object で、

- 現在の入力モードの表示、切り換え
- コード表からの入力

を skk.so との通信によって行なう。AdddictAux は辞書への単語登録を担当する。

図 14: PaletteAux

図 15: PaletteAux(文字コード表)

図 16: AddDictAux

4 考察

im-sdk を利用して LE を開発、利用するにあたって特に問題となりそうな事項について述べる。

4.1 セキュリティ

4.1.1 ユーザ認証

im-sdk では、現在、ユーザの認証方式として

- PAM を利用したパスワード認証
- libwrap を利用したアクセス制御
- 設定ファイルによるパスワード認証
- 設定ファイルによるアクセス制御

を利用する事ができる。しかし、LE 側からは、ユーザ名、ホスト名、IIIMCF のタイプ、といったごくわずかな情報のみしか得る事はできず、上に記したどの認証方式で認証されたのか、といった情報は入手できない。このため、ユーザは htt_server が動作しているシステムにアカウントを持っているかどうか、といった細かい制御ができなくなっている。

4.1.2 Loadable Module

htt_server は起動時にすべての LE Module を読み込む。その際に、一つでも動作しない LE が存在すると htt_server そのものが動かない状態におちいってしまう。また、各 LE が干渉する、といった相性問題についても考慮しなくてはならない。

4.2 ロカール機構

im-sdk では、ロカールに依存しない形で入力メソッドを提供する。LE が IIIMSF である htt_server に対するローダブルモジュールとして生成する関係上、LE を開発するにあたっては、Linux で整備されつつある国際化の枠組を利用する事ができない。IIIMF-SKK を作成する段階においては、ロカールに依存した文字列処理関数を用意する事でこの問題を回避しているが、その際にある程度文字コードについての知識が必要となる。

4.3 利便性

現状、Linux 上で IIIMF を利用する場合、IIIMF の機能を十分に利用するためには、LI18NUX で提供されている Xlib-I18N を利用しなくてはならない。一般的に利用される Linux Distribution では、Xlib-I18N が組み込まれていない事が多いため、IIIMF を利用する環境を整えるためにある程度の知識が必要となる。

5 今後

5.1 IIIMF

IIIMF は、現在、LI18NUX において次期バージョンの仕様の策定、実装が行われている最中である。予定されている項目は多岐に渡るが、主なものを以下に述べる。

5.1.1 PCE/EMIL

IIIM クライアントが IIIMSF と頻繁に接続しなくてすむための手段として、IIIMSF 側からキーボードからの入力を文字に変換するためのルールをクライアントに送信しておく、という手段がある。この機構を実現させれば、ネットワーク上での通信を必要最小限度におさえる事が可能になるが、Java 言語のようなネットワーク上でのセキュリティが考慮されていないような言語においては、それを実現する事は困難である。現在、LI18NUX では、XML を利用してセキュリティを確保しつつ変換ルールを定義する言語として PCE/EMIL が開発されている。

5.1.2 主要コンポーネントのライブラリ化

現在配布されている im-sdk では、独自に IIIMP を利用して通信するアプリケーションやサーバを作成する事が困難な状況であった。現在 LI18NUX では、IIIMP を利用して通信するためのライブラリを実装中である。これらのライブラリの実装が公開されれば、IIIMP を利用するソフトウェアを実装しやすくなる。特に、コンソール上で利用するアプリケーションや、gtk+-2.0 系列で採用された immodule などの機構に組み込む、といった事や、各言語への binding 等も行う事が容易になるため、開発速度が大幅に向上する事が期待できる。

5.2 IM Library

LibSKK や IIIMF-SKK の実装を通じて、入力メソッドを実装するために必要な要素は

- キーの入力を受けつけ、適切な振舞いを行う
- 辞書サーバへの問い合わせ

- 前編集が必要かどうかの判定と前編集領域の制御
- 設定ファイルの読み込み

といったある程度定まったものである事がわかってきた。これらの要素はある程度共通化する事が可能である。そこで、筆者は glib の version 2 系列で導入された GObject を利用して、IM に必要な要素の構築を容易にするためのライブラリを開発する予定である。また、すでに開発している LE についても、このライブラリを利用して再構築する予定である。

5.3 X Auxiliary Library

X Auxiliary Object として最低限必要な事項も、ある程度の共通項が存在する。実際、IIIMF-SKK のために作成した X Auxiliary Object は、ほとんど変更する事なく他の LE にも応用できた。現在の実装では、コード表のテーブルが EUC-JP エンコーディングに特化したものであったり、やみくもに実装した結果中身がすっきりしていない状況にある。これをより簡潔で汎用的にした形にし、更に LE の設定部分に関する UI を追加した形でライブラリ化する予定である。

5.4 SKK 以外の入力方法のサポート

現状では、IIIMF-SKK 以外に、LI18NUX から配布されている CannaLE と IIIMF-SKK をベースに、IIIMF-Canna を開発中である。この LE は近日中に公開する予定である。また、これ以外に、IIIMF-Wnn、IIIMF-POBox 等、様々な LE の開発を行う予定である。

6 まとめ

本論文では、IIIMF を利用する事で、OS 非依存、Window System 非依存の入力メソッドを提供する事ができ、LEIF を実装したモジュールを作成する事で IIIMF を利用した入力メソッドを提供する事が可能である事を示した。また、LEIF を実装した LE の一つである IIIMF-SKK の実装方法について述べ、LEIF を作成する上での考慮すべき事項を明らかにした。

IIIMF は、入力メソッドの SDK として利用する場合、非常に容易に扱う事が可能であり、比較的短期間で入力メソッドの開発を行う事が可能である。現在、日本語以外の言語においてもいくつかの LE が開発途上にあり、今後、IIIMF で利用する事が可能な言語も更に充実していくと考えられる。

一方で、IIIMF は、まだ発展段階にあり、改善すべき余地も残っている。IIIMF を利用するユーザもまだまだ少ないのが現状である。早期にユーザが利用しやすい状況になる事を目標に開発を進めていきたい。

```
static void
listener (SkkBuffer *buf, gchar *value, gpointer user_data)
{
    g_message ("listener");
    if (value) {
       g_message ("%s", value);
    }
    return;
}

int
main (void)
{
    SkkBuffer *buf;
    SkkFunc *func;
    SkkQuery *query;
    gchar tmp[8192];
    int i;
    buf = skk_buffer_new ();
    func = skk_func_new ();
    query = skk_query_new_with_path ("./.libs");
    skk_query_add_item (query, skk_query_item_new (SKKQUERY_LOCAL);
    skk_query_add_item (query, skk_query_item_new (SKKQUERY_SERVER);
    skk_query_add_item (query, skk_query_item_new (SKKQUERY_LOOK);
    skk_buffer_add_preedit_listener (buf, listener, NULL);
    skk_buffer_add_result_listener (buf, listener, NULL);
    skk_buffer_set_query (buf, query);
    skk_keymap_do_func (buf, func, SKK_VK_q, SKK_CONTROL_MASK);
    while (fgets (tmp, 8192, stdin)) {
       for (i = 0; tmp[i]; i++) {
      skk_keymap_do_func (buf, func, tmp[i], SKK_ALL_MASK);
       }
    }
    return 0;
}
```

図 10: libskk を利用したプログラム例

```
<?xml version="1.0" encoding="EUC-JP"?>
<!DOCTYPE iiimf-skk [
<!ELEMENT iiimf-skk (item,rule)*>

<!ELEMENT item (name, value, info)>
<!ATTLIST item type (bool|string|num) "bool">
<!ELEMENT name (#PCDATA)>
<!ELEMENT value (#PCDATA)>
<!ELEMENT info (#PCDATA)>

<!ELEMENT rule (key, katakana, hiragana, append)>
<!ELEMENT key (#PCDATA)>
<!ELEMENT katakana (#PCDATA)>
<!ELEMENT hiragana (#PCDATA)>
<!ELEMENT append (#PCDATA)>
]>
```

図 11: 設定ファイルの DTD

参考文献

[1] Masaki Katakai
Language Engine Interface Specification 1.2, 1999. http://www.li18nux.org/subgroups/im/IIIMF/spi/LEI.html

[2] Hideki Hiura
Internet/Intranet Input Method Architecture, 1999. http://www.li18nux.org/subgroups/im/IIIMF/whitepaper/whitepaper.html

[3] Hideki Hiura, Hidetoshi Tajima
Internet-Intranet Input Method Protocol(IIIMP) Specification Revision 2.0 Draft 0.2, 1999. http://www.li18nux.org/subgroups/im/IIIMF/protocol/spec.html