情　報　学　基　礎　55－2
デジタル・ドキュメント　19－2
（１９９９．７．１６）

文書交換における外字問題の一考察

道坂　修　　橋口　俊彦
株式会社 NTT データ　技術開発本部
{dosaka, hashi}@rd.nttdata.co.jp

近年に見られるインターネットの普及に伴い、インターネットを介して文書交換を行うニーズが高まっている。ところが、文書中に含まれる文字コードの独自性や交換を行うシステム間のプラットフォームおよび言語の差異より、文字化けなどの文書交換の真正性が保証されないケースが増えている。本稿では、文字セット、文字コードおよびフォントの観点から、文書交換において文書の真正性が保証されるための条件を考察し、特に文書に外字が含まれる場合に起こりうる問題を想定し、これら問題に対する対処方法を考察する。

Consideration of user defined characters in document exchange situation

Osamu Dosaka　　Toshihiko Hashiguchi
Research and Development Headquarters,　NTT DATA Corporation.

With the recent spread of Internet, document exchange systems are required in many cases. But because of the uniqueness of character code in documents and the difference of platform and/or language on each systems, many case result in failure to assure the documents are correct. In this paper, viewed in character set, character code and fonts, we consider the requirement of correct document exchange. Especially taking the problem of user defined character into consideration, we make a study of solution about above problems.

1. はじめに

近年の WWW の急速な普及により、インターネットを介して情報交換を行うシステムのニーズが高まっている。例えば企業間では CALS、EDI、EC などの企業間情報流通システム、官公庁では霞ヶ関 WAN を介した公文書交換システム、また総務庁の提唱する行政情報化推進共通実施計画[1]の一つの施策である電子申請システム等が挙げられる。特に上記の電子申請システムの場合、各省庁間で申請書類などの文書を交換する必要があるが、各省庁の行政情報システムは、主にホストコンピュータで管理されており、そのプラットフォームは各省庁間で統一がとれていない。また、行政情報システムの多くは、主に人名や地名においてプラットフォームの提供する文字セットで表現できない文字に対し外字を定義することで運用してきた。このような背景より省庁間の文書交換では省庁間のプラットフォームの差異や人名・地名に使用される外字によって文書交換の真正性が保証されない恐れがある。

本稿では、文字セット、文字コードおよびフォントの観点より、文書交換において文書の真正性が保証されるための条件を考察し、特に交換が困難と言われる外字の問題について考察を試みる。なお、本稿における考察は日本国内における文書交換を対象とし、日本国外に対する文書交換や ISO-10646/Unicode の持つ多言語処理の問題は論じないものとする。

2. 文字の管理モデル

2.1. 文字コードによる文字概念の管理

コンピュータに用いられる情報交換用図形集合(文字セット)やそれを符号化する方式(文字コード)は、国際標準化機関(ISO)および日本工業規格(JIS)で規格化されている。国内の標準規格文字セットは、JIS-X-0201、JIS-X-0208、JIS-X-0212、そして現在規格化作業を進めている新たな文字セット JIS-X-0213 がある。

これら規格における「文字」とは、規格の定める包摂規準に従い印刷字形の差異を包摂した文字そのものが表す概念を意味しており、規格表における個々の文字に対する印刷字形はその概念を表現する形状として最も標準的である字形を表している。従って、印刷字形の差異を概念の差異、すなわち別の文字と捉えこれらを区別するシステムにおいて、規格で規定される包摂規準に従った文字コード表現によりこれら異体字を区別することができない。

また各プラットフォームでは上記規格に加え独自の文字セットとそれを符号化する独自の文字コードを実装している。このプラットフォームが独自にサポートする文字セット(メーカ定義文字セット)の選定は各メーカの独自の判断であり、規格の定める包摂規準に矛盾する文字が別の「概念」として文字コードで表現されるケースがある。例えば、包摂規準で「高」に包摂されるべき文字である「高」が Windows では拡張文字セットに含まれ、シフト JIS 上では別コードが割り当てられている。このように、文字における包摂規準によりその印字字形が統合されているものとそうでないものが存在し、混乱を招いている。

2.2. フォントによる図形情報の管理

一方、文字コードで表現される文字をデバイス上に表示する場合、プラットフォームでは一般にフォントと呼ばれる図形情報を元に文字の描画処理を行う。

これについて注意しておきたいこととして、字形の表現はフォントに委ねられている点が挙げられる。例えば国語審議会の表外漢字字体表試案[2]では、「森鴎外」の「鴎」などの表外文字 39 文字について康熙字典字体に基づいた印刷字形を規定しており、この試案を採用するワープロやプリンタなどが増えていくことが予想される。この場合、文字コードとしては同一でもプラットフォームによっては表現される字形が異なるような字形のゆれが発生することが懸念される。また年賀状ソフトなどに添付されるフォントには書体に古風な味付けを加えるために書体のアレンジを施すばかりか、以下のように字形まで変更しているものも存在する。

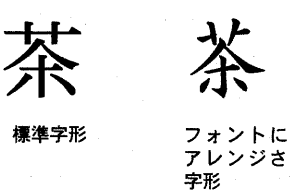

図 1　字形のゆれの例

上記の現象は、前述の文字コードによる文字表現が概念であるため、これを表示する場合包摂規準によって統合される印刷字形のどれを使用しても良いために生じている。

2.3. 文字概念と字形の分離

このように、プラットフォーム上では文字概念の管理と字形の管理を分離しているために、字形表現の差異に対する解釈が文字コードによるものとフォントによるものとで異なる[3]。従って、文書交換において、字形の差異を表現上と捉え概念上では同一であるというパラダイムでは、交換する文書に字形を再現するフォントを添付すると

いう条件付きで現状の仕組みで十分であるが、包摂規準に含まれる字形の差異を表現のみならず概念上でも区別したい場合、新たに文字概念を作成する必要がある。

3. 文書交換の真正性保証の条件

3.1. 文書交換の真正性の定義

以上を踏まえると、文書交換の真正性が保たれることの定義として、以下の二つの解釈が考えられる。

定義 1:文字の意味情報が正しく交換される
定義 2:文字の表示情報が正しく交換される

定義 1 では、文字コードなどの概念情報によって交換元および交換先で意志疎通を図ることができることを意味する。

定義 2 では、デバイス上に表現される字形の形状が交換元および交換先で同一であるレベルを指す。

3.2. 文字の意味情報の交換

定義 1 では、各プラットフォームはそれぞれサポートする文字コードが異なるため、文書を交換する際に各プラットフォームのサポートする文字コードに変換する必要がある。しかし現実的には、サポートする文字セットの範囲が異なることや外字が存在するため、すべてのプラットフォーム間で相互交換ができるとは限らない。例えば、WindowsNT 上で Unicode によって表現される JIS 補助漢字を含むテキストファイルを Windows3.1 でシフト JIS に変換する場合、シフト JIS は補助漢字をサポートしていないため、コード変換が不可能である。従って、定義 1 の条件を満たすためには、交換先と交換元で文字セットが共有できなければならない。

また、文字コード変換を行う際に、コード変換アルゴリズムを決定するために交換する文書の文字コードが何であるのか理解できることが必要である。ISO-2022-JP や日本語 EUC、シフト JIS などの標準的文字コードは、バイナリデータの特徴より比較的容易に判別することができるが、メーカ定義の文字コードや Unicode、UTF-8、UTF-16 などの他多言語文字コードの混在を考慮すると交換先で文書中のこれら文字コードの判別を行うことは不可能に近い。従って文書交換を行う際に文書を記述している文字コードが何であるのか、交換元で明示してやる必要がある。交換する文書の内部に文字コードを明示するアプローチとしては XML が挙げられる。XML では、デフォルトに使用される文字コードは UTF-8 および UTF-16 と決まっており、これ以外の文字コードで記述した場合、XML 宣言内にそのエンコーディング方法を含めなければならない。また http では HTML 文書の要求に対し、サーバ側で Content-type を出力する際に charset パラメータを使用[4]してブラウザに HTML 文書の文字コードの種別を明示している。しかし、いずれのアプローチも標準規格の文字コードのみを対象としており、メーカ定義の文字コードの場合は判別できない。

従って定義 1 の条件を満たすためには、メーカ定義文字コードを含め、交換元で文書に使用されている文字コードの種別を理解できる必要がある。

3.3. 文字の字形情報の交換

定義 2 では、字形の表示はフォントに依存するため、文書におけるフォントの指定が必須となる。従って、フォント情報を持たない文書やテキストデータを交換する場合、字形を特定する情報は文字コードのみに依存するため、定義 1 の条件が必要となる。また HTML や各種ワープロ文書のような書式情報の付いた文書を交換する場合は、交換元で使用したフォントが交換先のシステムにインストールされている必要がある。例えば HTML に CSS のスタイル定義や一部のブラウザ

による FONT タグの拡張によってフォントを指定した場合、これを参照するブラウザの動作する端末上にその指定したフォントがインストールされていない限り HTML 作成者の意図する字形を表示することができない。

これに対し、交換する文書そのものにフォント情報を含める（フォント埋め込み）アプローチがとられつつある。従来の HTML に見られる文字の印字字形をイメージとして添付する方法や、OpenType[5]や TrueDoc[6]に見られるフォントファイルの埋め込みなどが挙げられる。

3.4. 文書交換の真正性保証の条件

上記二つの定義に要求される条件をまとめると以下のようになる。

条件 1:交換する文書に使用されている文字セットを交換元で理解できること

条件 2:交換する文書に記述されている文字コードを交換元で判別できること

条件 3:交換先および交換元でフォントを共有できること

上記条件において、特に条件 1 では交換先で理解できない文字セットは外字として扱われるため、外字に対する対策を考慮する必要がある。

4. 外字の定義

一般に外字と呼ばれる文字の定義は以下の二つに分かれる。

(1) プラットフォームのサポートする文字セットに含まれない文字

(2) 標準規格文字セットに含まれない文字

図 2 にプラットフォームのサポートする文字セットと標準規格文字セットとの関係を示す。同図にて、全体文字セット Z と標準規格文字セット X およびプラットフォーム文字セット Y との包含関係より定義される文字セット A,B,C,D は

文字セット $A = \overline{X \cup Y}$

文字セット $B = X \cap Y$

文字セット $C = X \cap \overline{B}$

文字セット $D = Y \cap \overline{B}$

を表す。

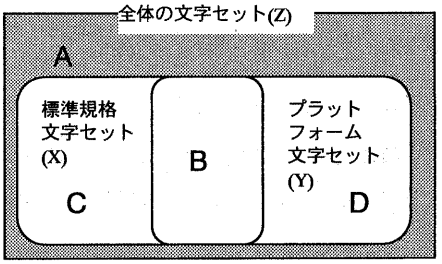

図 2 文字セットの分類

これより、外字の定義(1)(2)は

(1) プラットフォームのサポートする文字セットに含まれない文字... $\overline{Y}$

(2) 標準規格文字セットに含まれない文字... $\overline{X}$

であるため以下のように表すことができる。

(1) $\overline{Y} = A \cup C$

(2) $\overline{X} = A \cup D$

上記式より、外字であることの定義として、プラットフォームのサポートするに含まれない文字セットである場合も、標準規格文字セットに含まれない文字セットである場合も、共通して文字セット A が含まれ、両者の定義の違いは文字セット C と D であることがわかる。

4.1. メーカ定義文字

図 2 の文字セット D はメーカ定義文字と呼ばれる。例えば JEF コードは文字セットとして JIS-C-6226-1978(旧 JIS)をベースに、JEF 拡張文字と呼ばれる約 5000 ものメーカ定義文字を含んでいる。また日本語版 Windows(NT を除く)においても、JIS-X-0208 をベースに Microsoft 漢字と呼ばれるメーカ定義文字が含まれている。これらメーカ定義文字は、プラットフォームにフォントを含めてデフォルトとして組み込まれているため、同

一プラットフォーム内でこれらメーカ定義文字を含めた文書交換を行った場合、前節の定義より文書交換の真正性は保証される。また、メーカ定義文字セットは、各プラットフォームで既知であるため、プラットフォームの異なるシステムへの交換は、文字コード変換表およびフォントの整備により実現可能であるが、すべての組み合わせに対しこれらのリソースを整備するには膨大なコストがかかるであろう。またプラットフォームの異なるシステム間で文字コード変換を行う場合、交換先で交換元のメーカ定義文字を扱えるようにするために交換先で新たに外字を定義する必要があり、交換元の文字コードのユーザ定義文字領域が不足する可能性がある。例えば、富士通系ホストコンピュータと Windows 端末間で上記の文字コード変換を行う場合、JEF コード[7][8]からシフト JIS への変換が必要となる。ところが、上記の場合JEF拡張文字分(5000文字)を新たにシフト JIS のユーザ定義文字領域に登録する必要があるが、シフト JIS に登録できる外字の数は 1880 であるため、ユーザ定義文字領域の領域不足となる。(旧 JIS と新 JIS の変換については説明を省略する。)

4.2. サポート外標準規格文字セット

図 2 の文字セット C は、標準規格文字セットでありながらプラットフォームのサポートする文字セットに含まれないものを表す。JIS 補助漢字は規格化された時期が最近であるため、Unicode を採用しているものを除くほとんどのプラットフォームはサポートしていない。また国外の文字セットなどもこれに含まれる。文字セットは既知であるが、メーカ定義文字と同様にこれを交換先システムで利用できるようにするためには、サポート外標準規格文字セットを外字として登録する必要があり、メーカ定義文字の場合と同様のユーザ定義文字領域不足の問題に陥る。

4.3. ユーザ定義文字

図 2 の文字セット A はユーザ定義文字と呼ばれる。ユーザ定義文字はメーカ定義文字とは異なり、システム提供時にはデフォルトとして定義されるものではなく、各端末で個別に作成される。従って、同一プラットフォームにおいても各端末で文字コード上の定義が異なり、文字セットも未知であるため交換が極めて困難である。

ユーザ定義文字は以下の種類に分類できる。

(1) 他プラットフォームのメーカ定義文字

これは前述のメーカ定義文字セットを交換するために定義される。

(2) サポート外標準文字セット

これも主に交換用として定義される。

(3) 修飾文字・合成文字

「㈱」や「㌔」(例はメーカ定義文字である Microsoft 漢字) など本来ワープロソフト等でレイアウト情報で記述されるべき文字を入力の利便性やスペース上の都合により定義される。また一部のシステムではテキスト入力エリアに入力できる文字数が仕様上の問題により限られていたため、外国籍などのテキストエリアの入力できる文字数を越す比較的長い文字列を入力する際にカタカナを合成して外字を定義した例もある。

(4) 異体字

標準の文字コードでは包摂規準により包摂される同義であるが印刷字形の異なる文字である。これを包摂する内字 (親字) に対して表現上だけでなく文字概念として区別したい場合に定義される。

(5) 重複して定義される文字

プラットフォームの提供する文字セットに重複して定義される文字である。使用者が IME などの入力支援プログラムより入力したい文字が見つからない場合に、内字として存在するにも関わらず外字として定義してしまうケースがある。

(6) その他
古文書で使用されていた旧字体や出典が確認できない文字（正字ではない文字）、俗字などがある。

4.4. 外字を含む文書を交換する際に生じる問題

前述の外字は、そのプラットフォームでサポートされるメーカ定義文字を除き、プラットフォームに標準で提供される外字エディタなどのユーティリティにより文字コード内のユーザ定義文字領域に割り当てられる。プラットフォームが異なる場合、ユーザ定義領域に割り当てられている外字を交換することは不可能であるが、同一プラットフォームでは文字コードの変換を必要としないため、そのまま交換できる。この場合、交換先と交換元で同一のコードポイントに同じ文字が割り当てられているとは限らないため、異なる文字として解釈される恐れがある。例えば、シフトJISのユーザ定義領域0xF040に「邊」を割り当てているWindows端末から、同一のコードポイント0xF040に「薫」を割り当てているWindows端末にコードポイント0xF040の文字を含む文書を交換する場合、交換元では「渡邊」であった単語が交換先では「渡薫」に解釈されてしまい、意思疎通に支障が生じる。

5. 外字を含む文書の真正性保証

前節3.1で挙げた定義2を満たすためには、書式付文書の場合には、文書にフォントを埋め込み、交換を行うことで解決できると思われる。
交換する文書が書式を持たない場合や前節3.1で挙げた定義1を満たすためには、前節3.4で述べた条件に従い、交換を行うシステム間で外字情報（文字概念、フォント）を共有する必要がある。
本稿では外字の文字セットおよびフォント共有の実装について二つの方法を考察する。

5.1. 統一文字コードおよび文字参照による方法

交換する文書に使用する文字コードを統一し、これら文書を作成、参照できるアプリケーションをプラットフォーム毎に用意する。これにより、プラットフォームのサポートする文字コードからの制約から解放され、文字コード変換や文字コード判別の問題は解消する。また統一コードとして採用する文字コードは国内規格をはじめとする各種標準規格文字セットを広くサポートするUTF-8やUTF-16とする。これを採用することにより利用できる内字が増え外字登録の可能性が低くなることや前節4.2で述べたサポート外標準規格文字セットがなくなるため、これによる外字定義の必要がなくなる。またメーカ定義文字およびユーザ定義文字の文字コード表現は、従来のユーザ定義領域を使用することによって生じる外字登録数の制約を考慮し、JEPA[9]の出版データ標準化活動で提唱している外字表記方法[10]のような文字参照を使用することが考えられる。上記アーキテクチャを実現するためには、以下の仕組みを新規に導入する必要がある。

- 文字参照を解釈するパーサ
- メーカ定義文字や日々更新されるユーザ定義文字の字形および文字コードへの割り当てを管理するネットワーク型外字管理システム
- 外字フォントを提供するフォントサーバ
- フォントサーバから得られるフォントをラスタライズするレンダリングエンジン

また、文書交換元および交換先ではUTF-8およびUTF-16のサポートするすべての文字セットに対応するフォントを持つことは現実的に不可能であるため、これらフォントがインストールされていない端末における内字の表示は多言語フォントサーバ[11]などの枠組みを利用することで対応する。

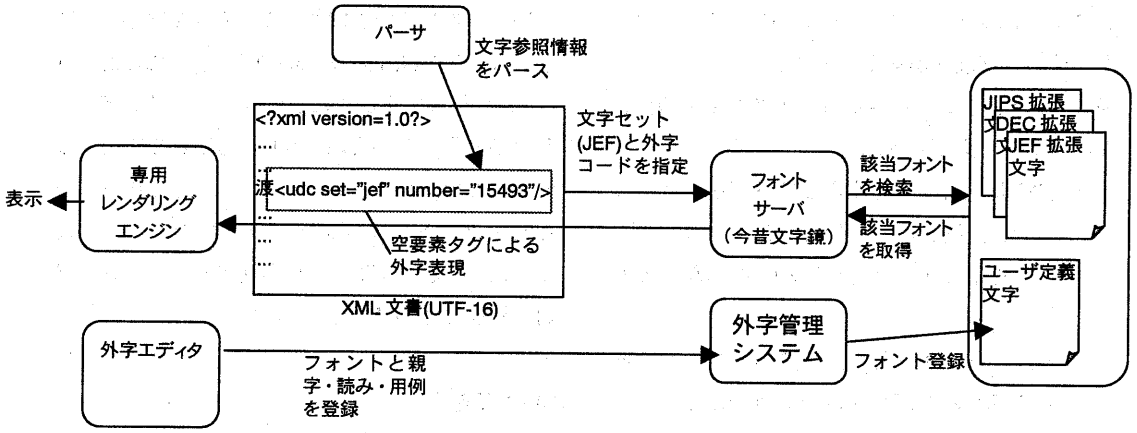

図 3　文字参照による外字表現

さらに、異体字などの印刷字形の差異を区別せず意味が同義である文字を検索したい場合、Unicode Version3 で実装予定である Variant タグ（異体字タグ）による親字の指定や外字管理サーバの管理する VACS モデルより同義である文字を展開し検索することができる。

この実装方式の問題点として以下が挙げられる。

- 上記仕様を満たすアプリケーションを各プラットフォーム毎に用意するのにコストがかかる。
- 既に運用されているシステムの遡及データの変換にコストがかかる。
- ワープロファイルなどのアプリケーションに依存するファイルでは統一コードに移行できない。
- 既存データベースとの親和性が悪い。

5.2. コード変換/外字セット切り替えによる方法

統一コードを用いずに、文書交換システムを用いて交換先プラットフォームの文字コードに変換する。文書交換システムでは、交換元および交換先プラットフォームの文字コード種別を指定し、各プラットフォームサポート文字コードの種別に応じた文字コード変換を行うため、文字コード変換テーブルの整備が必須となる。また前節で指摘したコード変換によるユーザ定義文字領域の不足の問題は、従来の外字表現に加え、コード変換用外字表現をエスケープシーケンス、文字セット情報、外字コードの組み合わせとし領域を拡大することで解決する。エスケープシーケンスは、プラットフォームにおいて使用されないコードポイントを選択することで決定づけるものとし、エスケープシーケンスで始まる 3 オクテットのバイト列をコード変換用外字 1 文字として扱う。文字セット情報は外字の文字セットを切り替えるために用いる。外字の文字セットを切り替える外字管理システムとして XKP[12]が挙げられるが、本方式は遡及外字データとの互換を考慮した外字コード表現である点やプラットフォームの異なるシステム間で文書交換を考慮している点でそれと異なる。

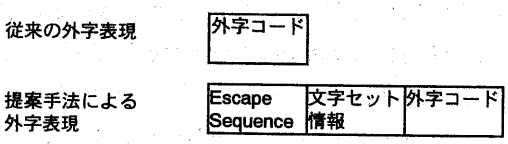

図 4　外字セット切り替えによる外字表現

また、外字コードおよびフォントは統一コード方式と同様に外字管理サーバおよびフォントサーバによって管理されるが、この方式では外字管理

システムはプラットフォーム毎にその外字コードの体系を管理する必要がある。

従って、上記方式を実現するためには以下の仕組みを追加する必要がある。

- 外字管理システムと連携して外字を含めた文字コード変換を行う文書交換システム
- 交換用外字表現(3 オクテット)を解釈するパーサ
- メーカ定義文字や日々更新されるユーザ定義文字の字形および文字コードへの割り当てを個々のプラットフォーム毎に管理するネットワーク型外字管理システム
- 外字フォントを提供するフォントサーバ
- フォントサーバから得られるフォントをラスタライズするレンダリングエンジン

この方式のメリットとして、遡及データがそのまま利用できる点や既存データベースとの親和性が良い点が挙げられる。

また問題点として以下が挙げられる。

- 日々更新されるユーザ定義文字によりコード変換テーブルのメンテナンスが困難である。特に各プラットフォームで追加、更新されるユーザ定義文字の同定作業にコストがかかる。
- 文書交換時にコード変換するため、コード変換処理にコストがかかる。

6. まとめ

文字セット、文字コードおよびフォントの観点から文書の真正性が保証されるための条件を考察し、外字を含む文書交換の真正性を保証するための方式を考察した。本稿における二つの提案方式は、それぞれの長所、短所を踏まえシステムに要求される仕様に応じて使い分けると良いだろう。また、文書交換を検索や入力を含めて広く捉えた場合、本稿では触れていなかった Unicode そのものの持つ問題や JIS の新旧字体の差異など考慮しなければならない問題が他に多々あると認識している。また今後予定されている JIS 第三・第四水準規格や Unicode の Plane 2 の SuperCJK 追加など、これからますます文字セットおよび文字コードの多様化が予想され、今よりも増して文書交換の真正性保証が要求されるであろう。今後はこれら文字セットおよび文字コードの動向を踏まえつつ、文書交換の真正性を保証するシステムの詳細を検討していく予定である。

参考文献

[1] 行政情報化推進共通実施計画”
http://www.somucho.go.jp/gyoukan/kanri/990402b.htm

[2] 国語審議会:”第 21 期国語審議会「審議経過報告」”, 国語審議会,1998.6
http://www.monbu.go.jp/singi/kokugo/00000011/

[3] 織田,木戸,小田:”Unicode の最新動向”,情報処理学会デジタル・ドキュメント研究会資料,12-3,p24,1998

[4] 村田他:”charset パラメータの勧め: HTML における文字符号化スキームの明示方法”
http://www.fxis.co.jp/DMS/sgml/html_correct_charset.html

[5] http://partners.adobe.com/asn/developer/opentype/otover.htm

[6] http://www.truedoc.com/webpages/intro/

[7] Ken Lunde:”日本語情報処理”,p342~343,ソフトバンク,1998.3

[8] JEF の概念説明書
http://www.fujitsu.co.jp/people/kanda/jef/catalog/gainen.htm

[9] JEPA(日本電子出版協会)
http://www.jepa.or.jp/

[10] http://www.est.co.jp/ks/dish/jepagaiji/gaiji.htm

[11] 前田他:”クライアントにフォントを必要としない多言語 HTML 文書ブラウジングシステム”, 情報処理学会論文誌 Vol.39 No.3,p802~809,1998.3

[12] Windows 拡張漢字処理:
http://www.xkp.or.jp/index.html