文字データベースに基づく文字オブジェクト技術の構築

Developping of Character Object Technology with Character Databases

守岡 知彦 [1)]　　クリスティアン・ウィッテルン [2)]
MORIOKA Tomohiko　　Christian Wittern

1) 京都大学 人文科学研究所 附属 漢字情報研究センター（〒606-8265 京都市左京区北白川東小倉町 47 番地
E-mail: tomo@kanji.zinbun.kyoto-u.ac.jp）

2) 京都大学 人文科学研究所 附属 漢字情報研究センター（〒606-8265 京都市左京区北白川東小倉町 47 番地
E-mail: wittern@kanji.zinbun.kyoto-u.ac.jp）

ABSTRACT.
The CHISE (CHaracter Information Service Environment) project is a character processing system which is based on the proposed character object model. This model is based on character property databases instead of coded character sets. Currently the system consists of two subsystems: XEmacs UTF-2000 and a prototype of Topic Maps engine using Zope. XEmacs UTF-2000 is an extensible editor into which a character database has been embedded. Within XEmacs UTF-2000 each character is created as an object which is defined by a set of character-attributes. In order to achieve a higher expressive power, a topic map of characters based on the ISO TopicMaps standard (ISO/IEC 13250) is under development. For the maintenance of this topic map, the prototype of a topic map engine has been developed based on the Zope object database server. In addition to that, a database of glyph expressions using IDS sequences for the more than 70000 Chinese characters contained in ISO/IEC 10646-1:2000 has been developed.

1 背景

計算機における文字表現技術は計算機におけるさまざまなデータ表現の基盤であり、インターネットにおける多彩なコンテンツも文字表現技術に依っている。現在、計算機における文字表現はすべて符号化文字技術に基づいている。これは文字を対応する番号で表現する方法である。

この方法では文字と番号の対応規則である符号化文字集合（文字符号）を両者が共有している必要がある。さもなくば文字化けが生じる。また、符号化文字集合に含まれる文字は有限であり、そこに存在しない文字は表現できない。このため、従来、交換可能性をあきらめて外字を用いる方法や、検索などの符号化文字としての利便性をあきらめて画像を用いる方法が採られてきた。また、こうした問題を避けるために、文字符号の規格に多数の文字の追加を行う動きもあるが、これにも技術的・政治的問題がある。

このような問題を抜本的に解決し、日常的な文書のみならず歴史的な文書や将来生じる文書も表現可能で、かつ、各文書の永続性を保証するためには、符号化文字集合に依存しない次世代の多言語文書処理技術の確立が必要である。このためには文書表現におけるあらゆる要素を機械可読な方法で宣言的に記述可能にする必要がある。すでに、文字より大きな単位の要素に対しては、SGML/XML などにもとづくマークアップ手法が用いられているが、文字に対してもその性質を機械可読な方法で記述し、このように表現された文字知識に基づいて文字を処理することは有効な方法であると考えられる。

このような文字の機械可読な記述に基づく次世代文書処理技術を確立し、実用化するためには汎用的な文字データベース・システムおよびクライアント・システムの効率的な実現が欠かせない。また、これらの上で利用可能な文字データベースのコンテンツもなくてはならない。また、インターネットでの普及を考えるならば、こうした技術の少なくとも中核は自由に利用可能なプログラムとデータで構成される必要がある。

2 目的

CHISE (CHaracter Information Service Environment) プロジェクトの目的は、符号化文字技術の問題点を抜本的に解決するために、文字に関するさまざまな知識を機械可読な形で表現・蓄積し、こうした文字知識に基づいて文字を処理するアーキテクチャを開発することである。このために、文字データベースに基づく文字オブジェクト技術を提案し、その実証を目的に文字の表現・処理方式及びその実装と幾つかの文字データベースを開発した。

こうした文字処理アーキテクチャを実用化するためには、文字データベース・システムと、そのコンテンツとなる文字データベース、および、この文字データベースを利用する文書編集系が必要である。

文字データベースの構成法としては、ファイル入出力や文字の表示のような高速性が重要となる場合と、漢字のさまざまな性質や文字間の関係を記述する場合のように複雑な情報に対する記述能力が重要となる場合がある。このため、高速性が要求される分野を対象とした『UTF-2000 方式』と文字知識の大局的な記述・処理を目的とした Topic Maps [3] に基づく方式を提案し、前者に対しては XEmacs [9] を元にした多言語文書編集環境 **XEmacs UTF-2000** の開発を進め、後者に対してはオブジェクト指向 WWW アプリケーション・サーバー Zope [10] に基づくプロトタイプを開発することにした。また、これらを有機的に結合するために、XEmacs UTF-2000 から外部のデータベー

図 1: XEmacs UTF-2000

スを利用するための枠組を開発した。

文字データベースとしては、XEmacs UTF-2000 での文書処理に必要な情報を中心に基本的な情報を収録した XEmacs UTF-2000 用文字データベースの編集・校正を進めるとともに、漢字の部品組み合わせ構造を表現した文字属性データベースを開発した。これは、漢字を検索に有用であるだけでなく、フォント合成や部品の包摂規準による文字の包摂規準を制御、あるいは、分散型文字オントロジーを実現する上で重要となる特定の文字データベースに依存しない漢字表現の基礎としても重要であると考えられる。

3 XEmacs UTF-2000

符号化文字集合に依存しない文字表現技術の開発・検証のため **XEmacs-UTF-2000**（図 1）を開発した。これは対話型統合環境 **XEmacs** [9] を基にしている。

XEmacs やその基となった GNU Emacs [7] は Emacs Lisp 言語 [6] を使って機能を拡張することができ、実際に Emacs Lisp で記述されたさまざまなアプリケーションが作成され利用されている。例えば、GNU Emacs/XEmacs の中で電子メールやネットニュースを読み書きすることができるし WWW 頁を見ることができる。

XEmacs UTF-2000 は文字データベースに基づく文字オブジェクト技術の検証だけでなく、GNU Emacs, XEmacs の利点を継承した実用的な環境を目的に開発を行なった。

(1) UTF-2000 方式

符号化文字方式に基づかずに文字を表現すること、すなわち、固有の番号で同定することなしに文字を指し示そうとするとしたら、どうすれば良いだろうか？ おそらくその一つの方法は指し示したい文字の性質を列挙することだろう。例えば、「あ」を指し示すとするならば

用字系 ひらがな
音価 /a/

のようにすればよいだろう。[*1] 漢字の場合には、発音だけでは不十分である。例えば、「字」を指し示したい時に

用字系 漢字
音 /じ/

とすれば、「字」以外にも「時」や「次」など /じ/ という音を持つ全ての漢字が含まれてしまう。総画数を付け

用字系 漢字
音 /じ/
総画数 6

とすれば、「時」や「事」などは除外され、「字」や「次」や「耳」などの総画数が 6 画の /じ/ という音を持つ全ての漢字の集合に限定される。さらに部首「子」を指定すればさらに限定されていくだろうし、文字の構造「『宀』の下に『子』」を指定したり、字義を指定すればさらに限定されるだろう。

このように文字を属性の集合で表現することによって、符号化文字方式に依らずに文字（ないしは文字の集合）を表現することが可能である。また、指定する属性を多くしたり少なくしたりすることによって、表現する文字の包摂規準を細かくしたり粗くしたりすることができる。

このような属性に基づく文字表現においては、文字の性質に関する情報は文字属性として文字表現自体に含まれている。よって、特定の処理に必要な情報は文字属性として文字表現に含めておかなければならない。例えば、表示を行なうには文字の字形を示す情報が必要である。逆に、表示という処理を行なわないことを前提とした文字表現には、字形という属性は不要となる。

このように各文字をその文字が持つ属性の集合によって表現し、こうした文字の属性の集合をデータベース化し、それを参照しながら文字を処理する方式をここでは『UTF-2000 方式』と呼ぶことにする。UTF-2000 方式は文字に関する知識を直接プログラムするのではなく、データベース化してそれを参照する方法であるので、計算機が扱うことができる文字の種類や文字の概念は符号化文字集合の定義に束縛されない。使用する文字データベースを取り換えることによって、容易に文字の種類や概念を変更可能である。このような高度な柔軟性を持つ半面、多数の文字を扱う場合、多量の記憶量が必要となると考えられる。このため、文字データベースを柔軟性を損なうことなく効率的に扱うための機構が必要となる。

ところで、UTF-2000 方式自体は既存の符号化文字技術と対立するものではない。符号化文字集合の種類と符号位置は文字属性の一つと捉えることは可能であり、利用したい符号化文字集合を文字属性に含めることにより、それらの符号化文字集合の世界と情報交換を行なうことは可能である。

(2) 文字オブジェクトと文字表現

XEmacs UTF-2000 は UTF-2000 方式に基づき文字データベースを参照することによって文字を処理する。このため、文字データベースを利用しやすいように文字の内部表現を変更している。文字表現空間を 30 bit に拡大したため、同時に利用できる文字数は大幅に増えている。文字は、文字 id と呼ばれる文字オブジェクトの id で内部的に管理され、この文字 id を用いて文字オブジェクトを参照して、処理が行なわれる。

文字オブジェクトは文字属性の集合として定義され、固有の「文字 id」が割り当てられる。文字を定義するために XEmacs UTF-2000 では define-char という組込み関数を

[*1] 変体がなを考えた場合に、変体がなでない「あ」に限定したい場合にはこれでは不十分だが…

用意している。この他、文字や文字属性を扱うための機能を用意している。

(3) 外部データベース

UTF-2000 実装では処理対象とするすべての文字の知識を文字属性として保持しなければならないために、符号化文字モデルに基づく従来型の文字処理系に比べて多くの記憶資源を必要とする。

XEmacs UTF-2000 では文字属性データベースは define-char 形式の Emacs Lisp プログラムとして表現され、文字属性データベース全体を読み込んだ状態の記憶イメージをダンプした実行形式を作り、そのダンプされた実行形式を用いる。このため、XEmacs UTF-2000 のダンプ後の実行形式の大きさと元となった XEmacs-Mule の実行形式の大きさの差は文字属性データベースを保持するための記憶資源の大きさを意味している。

初期の XEmacs UTF-2000 は文字属性データベースの保持するための機構の効率化が十分でなかったこともあり、i386 アーキテクチャ上の Linux において当時の XEmacs-Mule の実行形式が約 10 MB であったのに対し、約 5 万字分の文字データを保持した状態で実行形式が 40 MB を越えるようになった。その後、文字データを保持する機構を改良し、記憶効率を向上したため、最近の XEmacs UTF-2000 では約 7 万字分の文字データを保持した状態で実行形式は約 27 MB となっている。

このように、主記憶上に文字属性データベースを保持する方法は多くの記憶資源を要するという点で問題がある。そして、通常の利用で必要となる文字は数百から数千であり、必要とする文字属性も限られているということを考えると、文字属性データベース全体をダンプするという方法は望ましくないと考えられる。

文字属性データベース全体をダンプする方法のもう一つの問題点は、UTF-2000 実装と文字属性データベースが不可分になってしまうことである。つまり、異なる UTF-2000 実装間で文字属性データベースが共有できないことを意味する。また、UTF-2000 実装の開発と文字属性データベースのメンテナンスは非常に性質の異なる作業であるにも関わらず、両者を統合した形でソースファイルを管理しなければならない。

このような問題点を解決するために、XEmacs UTF-2000 において外部の文字データベースを利用するための機構を開発した。これは外部の文字データベースから文字属性を必要な時に情報を獲得する (lazy-loading) ための枠組と、外部文字データベースの種類毎の実装からなる。現在のところ XEmacs の **database** 機能（Berkeley DB のような属性値を保持するための単純なデータベースに対する抽象）を利用した外部文字データベースだけが実装されており、Debian GNU/Linux (sid) における Berkeley DB Version 3 でその動作確認を確認した。

この実装（以下、『lazy-loading 版』と呼ぶ）と文字定義をダンプする方式の実装（以下、『dump 版』と呼ぶ）の実行形式の大きさを TM 5800 上の Debian GNU/Linux (sid) において比較すると、約 7 万字のの文字定義を持つ XEmacs 21.2.43 UTF-2000 の dump 版の実行形式の大きさが 27 MB (strip 後 22 MB) であるのに対して、lazy-loading 版の実行形式の大きさは 15 MB (strip 後 10 MB) となった。ちなみに、XEmacs 21.2.43（mule 付き）の実行形式の大きさは 10 MB (strip 後 6 MB) である。

lazy-loading 版の実行形式の大きさがなお XEmacs-Mule よりも 5 MB 程大きいのは、XEmacs-Mule から引き継いだ Emacs Lisp code において、coded-charset を鍵とした char-table が多用されているせいだと考えられる。XEmacs UTF-2000 では char-table は char-id-table で実装されており、coded-charset を鍵にして値を設定した場合、その coded-charset に属するすべての文字に対して値を設定するようになっているため、必要な記憶量が膨らむと考えられる。また、char-table は文字属性と異なり外部データベースに対応付けられていないため、lazy-loading ができないのである。よって、この点を改良すれば lazy-loading 版 XEmacs UTF-2000 の実行形式の大きさを XEmacs-Mule と同程度にすることができると考えられる。

4 文字属性データベース

(1) XEmacs UTF-2000 附属の文字データベース（UTF-2000 基本データベース）

これは XEmacs UTF-2000 のために開発している define-char 形式で表現される文字属性データベースである。これは、文字データベースに基づく文字オブジェクト技術の実証を目的とするとともに、将来における CHISE アーキテクチャに基づく文字情報交換のベースとなる標準的なデータベースとして利用することも視野に置いている。

現在のデータベースは

- Unicode Database
- CNS 11643 と諸橋大漢和辞典の対照表
- CDP (Chinese Document Processing) データベース
- CBETA (Chinese Buddhist Electronic Text Association) 外字データベース
- CHINA3 外字データベース
- 開発者がこれまで作成してきたその他の雑多なデータベース

等を統合し、互いの矛盾点を修正するものである。まだ誤りも多く、品質は高くはないが、現時点で約 7 万字分の定義が存在する。

この標準文字データベースでは、非漢字に関してはおおむね Unicode [8] の定義に則っている一方、漢字に関しては微小な字体差も区別している。漢字の各文字（字体）の内、大漢和辞典と同じ字体でないものについては、`morohashi-daikanwa` という属性の値として、大漢和番号と差異の度合および整理番号を持たせている。また、*The Unicode Standard* [8] の例示字体と同じ字体でないものに対しては、対応する Unicode の符号位置を `=>ucs` という属性の値とする。

(2) 漢字構造情報データベース

多くの漢字は偏と旁などの部品の組み合わせによって構成されている。こうした漢字の部品の組合せ構造は形の抽象的表現となるだけでなく、字義や音価にも関係しており、字源に基づく文字構造の分析は「解字」と呼ばれ、そうしたデータは重要な辞書記述の 1 つである。そこで我々は、UTF-2000 基本データベースに収録されている全ての複合（会意・形声）漢字に対し漢字構造情報を付けることを目標に、漢字構造情報データベースの開発をはじめた。

漢字構造情報に基づく符号化手法の試みは 1970 年代に遡るが、漢字符号化の主流とはならず、標準的な記法も確立されて来なかった。その後、ISO/IEC 10646-1:2000 [4] においてはじめて漢字構造情報の標準記法である IDS (Ideographic Description Sequence) とそのためのオペレーター群である IDC (Ideographic Description Characters)（図 2）が定義された。そこで、我々はこの IDS に基づく方法と、これを S 式（Lisp 表現）化し付加情報を許した *ideographic-structure* 形式、および、これを Topic Maps 化したものを用いることにした。

既存の漢字構造情報を含んだデータベースとしては、台湾中央研究院の CDP データベースと台湾の中華電子佛典協會 (CBETA) の外字データベースがある。これらはそれ

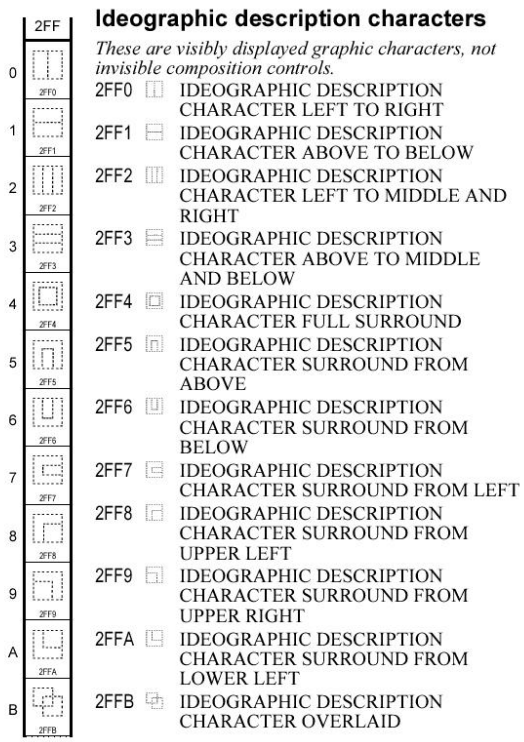

図 2: Ideographic Description Characters

ぞれ独自の形式を採っている。なお、これらは GPL で配布されるプログラムで利用可能であるので、可能な限り変換して利用することにした。また、この他、日本でも「今昔文字鏡」があり、検字を目的としたは 8 万字以上の解字情報を持つが、今の所、自由ソフトウェアで利用することはできない。またこの他、和田研フォントや GT 書体など、フォント合成を目的にしたいくつかの試みが存在するようである。

a) CDP データベース

CDP (Chinese Document Processing) データベースは、1990 年から台湾中央研究院の謝清俊らが開発している文字データベースで、現在、漢語大辞典に収録されている文字を中心に 55500 字以上を含んでいる。[*2]

CDP データベースは Big5 と外字を利用して符号化されており、外字を利用して 14 種類のオペレーターを定義している。そのうち 3 種類は部品の結合を表現したもので、(1) 縦に並べる、(2) 横に並べる、(3) その他 を表現する。また 8 種類の反復記号がある。これは同じ部品を複数個配置することを表現するものである。また、この他に特殊記号が用意されている。なお、CDP データベースの漢字構造表記では入れ子を認めていない。また、IDS に比べ、結合オペレーターの種類が少なく、結合オペレータ (3) から IDC への変換に曖昧性があるため、機械的に IDS へ変換することはできない。

b) CBETA 外字データベース

台湾の中華電子佛典協會 (CBETA) は仏典データベースの作成上必要となった外字を対象に文字データベースを作成しており、この外字データベースには現在 13000 字程が収録されている。

CBETA 外字データベースも Big5 と外字を利用して符号化されているが、オペレーターは ASCII 文字を利用する。中置記法を使っており、‘(’ と ‘)’ を用いることで入れ子表現も可能である。結合オペレーターとしては CDP データベースのものと同様に、縦に並べることを表現する ‘/’ と、横に並べることを表現する ‘*’ と、その他を表現する ‘@’ の 3 種類である。この他に、部品の削除と置換を表現するためのオペレーターが存在する。部品の削除は $A - B$ という式で表現され、これは文字 A から部品 B を削除したものを表現している。例えば、草冠は 草 − 早 で表現できる。部品の置換は $A - B + C$ という式で表現され、これは文字 A 中の部品 B を C に置き換えることを表している。例えば、「花」は 草 − 早 + 化 で表現できる。この削除と置換を用いることで、3 種の結合オペレーターだけでは十分に表現できないような漢字でも、多くの場合において曖昧無く表現可能である。次により複雑な例として (((瞭 − 目) − 小) − 日 + (工/十)) ∗ 支)/皿 を示す。CBETA 外字表現は小数の規則で驚く程多くの漢字をカバーできる半面、複数の式表現が生じやすいといえる。なお、なるべく ‘@’ を使わずに表現された式表現は、結合漢字の IDS 形式のデータベースが存在すれば、機械的に IDS へ変換することが可能である。

c) CHISE 漢字構造情報データベース

我々は漢字構造情報表現として、IDS に基づく入れ子上の木構造形式を採用し、プレーン・テキストでは IDS 形式、XEmacs UTF-2000 の文字データベースでは *ideographic-structure* 形式に基づく `ideographic-structure` 属性、XML では Topic Maps に基づく形式で扱うことにした。

そして、CDP 形式や CBETA 形式からの変換プログラムを作成し、機械的に変換可能な部分に関しては利用可能な既存のデータを用いることにした。しかしながら、前述のように CDP データベースは機械的に変換することができず、また、形式に則っていないと思われる部分もあり、我々の目的にとっては十分なものではなかった。一方、CBETA 外字データベースは仏典で用いられる特殊な文字が主であり、データソースの点で難点がある。また、IDS に変換するためには結合漢字を IDS で表現したデータベースが必要となる。そこで、CDP データベースから変換したデータを修正・補完することで、Unicode の基本統合漢字、ISO/IEC 10646-1 の統合漢字拡張 A, ISO/IEC 10646-2 [5] 統合漢字拡張 B, その他のレパートリの順に漢字構造情報データベースを開発することにした。

現在の所、基本統合漢字 3、拡張 A および拡張 B に関する入力作業はほぼ終了しており、現在、校正作業を行っている。また、この入力作業を行うために、CDP 外字や ISO/IEC 10646-1,2 の漢字を含む 7 万字以上の漢字を対象とした四角號碼方式の入力システムを quail（GNU Emacs/XEmacs 用の標準的な入力システムの 1 つ）を使って実現した。

5 Zope を用いた Topic Maps 文字知識データベース

文字は、異なる地域に分布するさまざまな書記言語のための用字系 (script) の中で運用される。これらの文字の正確な形は、音声言語における発音と同様に、時代や地域によって変化する。一方、UTF-2000 モデルは単一の文字を符号化文字集合に依存せずに表現するという点では有効なものの、文字の諸属性や文字間の複雑に入り組んだ関係を記述する上では問題がある。そこで、十分な表現力を有する別のモデルを用いて複雑な文字知識を記述し、これを UTF-2000 の世界と相互に変換可能にすることにより、高速性と表現力の両立を計ることにした。

文字知識の表現形式としては、現状を鑑みれば XML に

[*2] Christian Wittern は 1994 年よりこのプロジェクトに参加している。

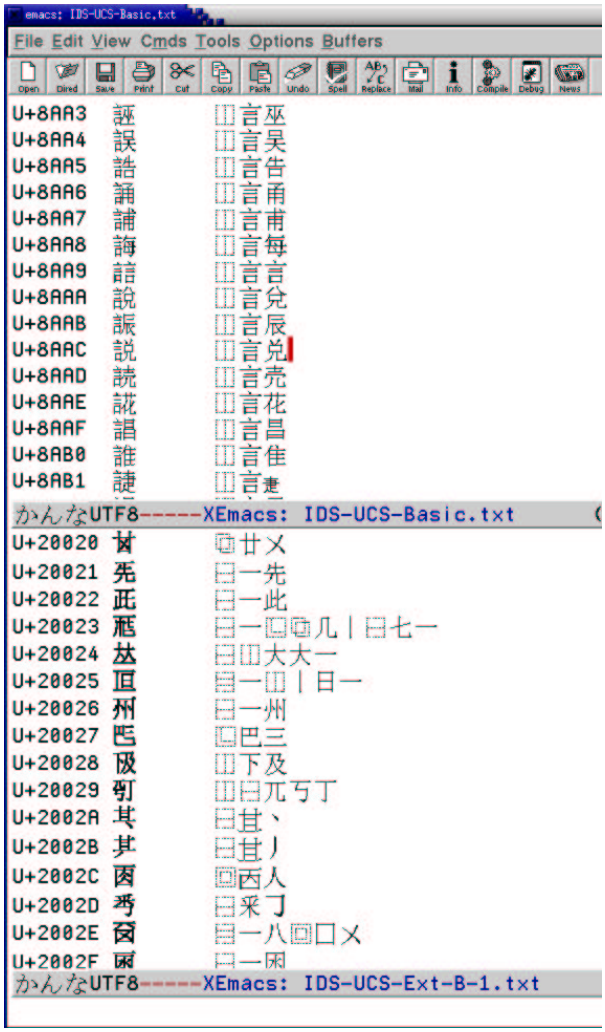

図 3: 漢字構造情報データベース

基づく手法を採用するのが適切であると考えられる。そこで、我々は、複雑に入り組んだ情報を SGML/XML に基づいて交換可能な形で表現するための標準的な記法のひとつである Topic Maps を用いることにした。これは対象となる情報の構造を topic (『話題』) というものとしてとらえ、topic の定義や topic 間の関係などを記述することで、さまざまな構造を持った各種情報リソースを表現しようとする ものである。

この国際標準で定義される記法を用いた 1 つもしくは複数の関連する文書の集合を 'topic map' と呼ぶ。一般に、topic map で伝達される構造を持った情報には

1) topic の近くに配置可能な情報オブジェクト群 (「出現」(occurrence))
2) トピック間の関係 (「連想」(associations))

が含まれる。

2000 年に出版された国際標準 [3] では、SGML [1] および HyTime [2] Architectural Forms に基づく形式の DTD (Document Type Definition) 表現による仕様が定義された。また、独立したヴェンダーのグループが XML 版の開発を開始し 2000 年 12 月に XTM (XML Topic Maps) 1.0 として公開した。そして、これは 2001 年 12 月には ISO 標準の修正 (amendment) として承認された。そこで、我々はこの XML 版に基づいて開発を行った。

(1) Topic Maps に基づく文字データベース

Character Topic Maps (CTM) は Topic Maps に基づいて文字知識を表現するための形式であり、現在、

- 抽象文字
- 文字インスタンス
- 異体字形
- 文字構造
- 言語
- 読み
- 意味
- 時代
- 空間
- 良く使われる用例
- 符号化文字集合への写像
- 辞書への参照

などの文字属性軸を定義している。

ここで例を示す。図 4 に define-char 形式の文字定義の一部を示す。

```
(define-char
  '((morohashi-daikanwa 35556 0 0)
    (ideographic-radical . 149) ; 言
    (ideographic-strokes . 7)
    (total-strokes       . 14)
    (ideographic-structure
     ((name . "IDEOGRAPHIC DESCRIPTION CHARACTER LEFT TO RIGHT")
      (ucs                . #x2FF0)     ; ⿰
      )
     ((ucs                . #x8A00)     ; 言
      )
     ((ucs                . #x514C)     ; 兌
      ))
    (chinese-cns11643-1 . #x6B29)       ; 說
    (ucs                . #x8AAA)       ; 說
    ))
```

図 4: U+8AAA の define-char 形式の文字定義

これを Character Topic Maps 形式に変換したものを図 5 に示す。これに見えるように、Lisp においてキー・値の対で表現されていた文字属性は、「出現」を表す `<occurrence>` 要素を使って表現される。なお、この図には、この Topic Maps を表現するための基礎となる構造は示されていないが、実際にはさまざまな topic を使った詳細な記述を行っている。紙面の都合上このようなものの全てを示すことはできないが、「出現」(occurence) として表記されていない `ideographic-structure` 属性の記述例に関しては示すことにする。図 6 に示すように、この属性は`<association>` 要素を用いて漢字部品の結合に関する別の topic を用いて表現されている。

(2) Zope を用いた Topic Maps エンジン

Zope (Zope Object Publishing Environment) は、Zope Corporation（旧 Digital Creations）によって共同体に基づくオープン・ソース・モデルによって開発されている、オブジェクト指向 Web アプリケーション・サーバーである。Zope は、C で書かれた少数のクリティカルな部分を除き、Python で書かれている。Zope は、通常、動的な Web コンテンツを素早く開発するための環境と考えられているが、本来はオブジェクト操作環境である。Zope を支えるストレージは、Topic Maps のような階層データ構造を格納するために設計された、オブジェクト指向データベースである。そして、Zope は Web サーバーとして動作するので、ネットワーク化されたデータベースと見なすこともできる。Zope は HTTP プロトコル、WebDAV および XML-RPC を通じてアクセスできる。このように、Zope に基づく実装を使うことの利点の一つは、分散型編集環境として利用できることであり、また同時に、XEmacs

```
<!-- character *說 start -->
 <topic id="_08AAA">
  <instanceOf>
   <topicRef
    xlink:type="simple"
    xlink:href="#abstract_character"/>
  </instanceOf>
  <subjectIdentity>
   <topicRef
    xlink:type="simple" xlink:href="
    #_idc-left-right_08A00_0514C"/>
  </subjectIdentity>
  <baseName>
   <baseNameString>*說
   </baseNameString>
  </baseName>
  <occurrence>
   <instanceOf>
    <topicRef
     xlink:type="simple"
     xlink:href="#hydzd-ref"/>
   </instanceOf>
   <resourceData>6.3979.3
   </resourceData>
  </occurrence>
  <occurrence>
   <instanceOf>
    <topicRef
     xlink:type="simple"
     xlink:href="#hydcd-ref"/>
   </instanceOf>
   <resourceData>11.239.3
   </resourceData>
  </occurrence>
  <occurrence>
   <instanceOf>
    <topicRef
     xlink:type="simple"
     xlink:href="#morohashi-nr"/>
   </instanceOf>
   <resourceData>35556</resourceData>
  </occurrence>
 </topic>
```

図 5: U+8AAA の Character Topic Map 表現（部分）

UTF-2000 からアクセス可能なバックエンドとして動作可能であることでもある。

Topic Maps に関する概念の幾つかはまだとても新しく、Topic Maps コミュニティの中でまだ十分に成熟していない（例えば、Topic Maps Query Language (TMQL) はまだ要求の段階にあり、topic map に対する問い合わせということが何を意味するのかについてまだコンセンサスを得ていない）。このため、より深遠な特徴の幾つかに関してはまだこのプロトタイプは対応していない。Topic Map の階層構造に対する問い合わせは、topic map から演繹

```
<!-- associations -->
 <association id="_idc-left-right_08A00_0514C">
  <instanceOf>
   <topicRef xlink:type="simple"
    xlink:href="#ids_with_2_members"/>
  </instanceOf>
  <member>
   <roleSpec>
    <topicRef xlink:type="simple"
     xlink:href="#idc"/>
   </roleSpec>
   <topicRef xlink:type="simple"
    xlink:href="#idc-left-right"/>
  </member>
  <member>
   <roleSpec>
    <topicRef xlink:type="simple"
     xlink:href="#first_component_character"/>
   </roleSpec>
   <topicRef xlink:type="simple"
    xlink:href="#_08A00"/>
  </member>
  <member>
   <roleSpec>
    <topicRef xlink:type="simple"
     xlink:href="#second_component_character"/>
   </roleSpec>
   <topicRef
    xlink:type="simple"
    xlink:href="#_0514C"/>
  </member>
 </association>
```

図 6: U+8AAA の漢字構造情報の Character Topic Map 表現

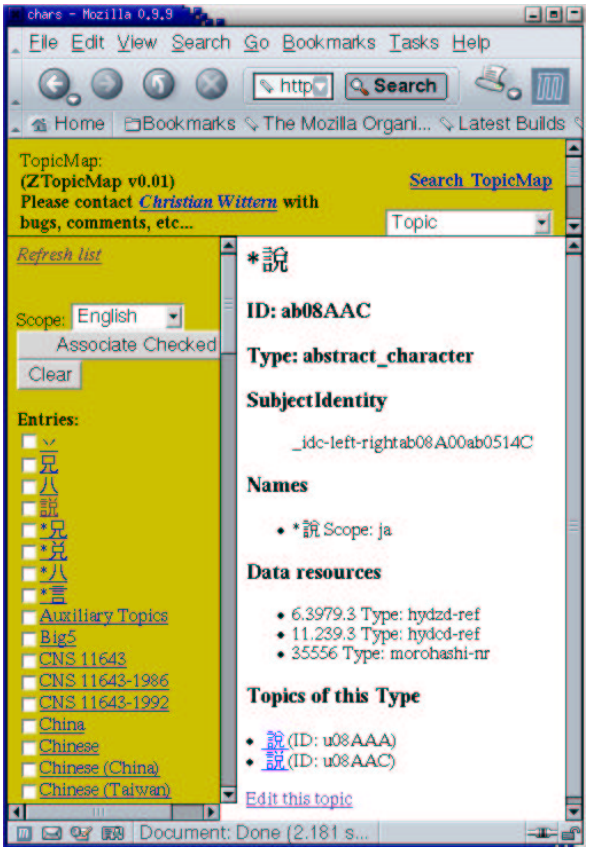

図 7: Zope に基づく Topic Maps エンジンのプロトタイプ

や他の Topic Map 計算に関わる複雑な処理を行う必要があるので、このプロトタイプでは topic map を構成する要素に対する単純な検索のみをサポートすることにする。同様に、複数の topic map のマージ命令は、‘Topic Map Basename Constraint’ (TMBC) で規定される topic の正規化に依存し、非常に複雑なので、これもまたこのプロトタイプではサポートしない。

結局、プロトタイプの開発において次の目標を設けた：

- XEmacs UTF-2000 からのデータのインポート・エクスポート
- 通信プロトコルに基づくネットワークの利用
- Topic Map に対するアクセス手段の提供
- 特定のデータ型に依存しない、汎用的な Topic Maps のための設計
- このアプローチの重軟性の評価ができること

図 7 に Zope を用いた Topic Maps エンジンのプロトタイプの動作例を示す。この実装は、Topic Maps エンジンとしてはまだ不完全である。幾つかの基本的な Topic Maps 処理に関してまだ問題があり、まだ解決に至っていないため、このエンジンはまだ開発段階のものといえる。

汎用的な Topic Maps エンジンの実現は、Topic Maps 標準が規定する仕様が複雑であるために、その開発はなかなか困難であることが判った。そのため、CHISE プロジェクトが必要とする文字知識データベースを実現する上で必要となる機能のみを実現する方が現実的であるといえる。よって、こうしたものを実用的な形で実現することが重要であるといえる。

一方、Zope は潜在的に非常に大きなデータを持つ Topic Map を保持するためのプラットフォームとしては適切ではないかも知れないことが明らかとなった。このため、よりスケーラビリティの高いデータベース処理系を使う方が

望ましいといえる。

現在の Topic Maps エンジンおよび XEmacs UTF-2000 との界面の実装モデルは、2 方向の通信に基づいている。

Zope オブジェクトデータベースに Topic Map を格納することはパフォーマンス上のボトルネックになるのは明らかである。この問題を解決するための良い方法はデータを外部ストレージに移動することであろう。このアプローチを実証するために、Topic Map データ構造を関係データベースの表の集合に写像し、関係データベース処理系のひとつである PostgreSQL から Topic Map エンジンにインポートするようにした。

このアプローチにおいて、XEmacs UTF-2000 との通信と Zope 内の Topic Map エンジンとストレージ・バックエンドは、図 8 に示すような 3 者間の関係に変わることになる。

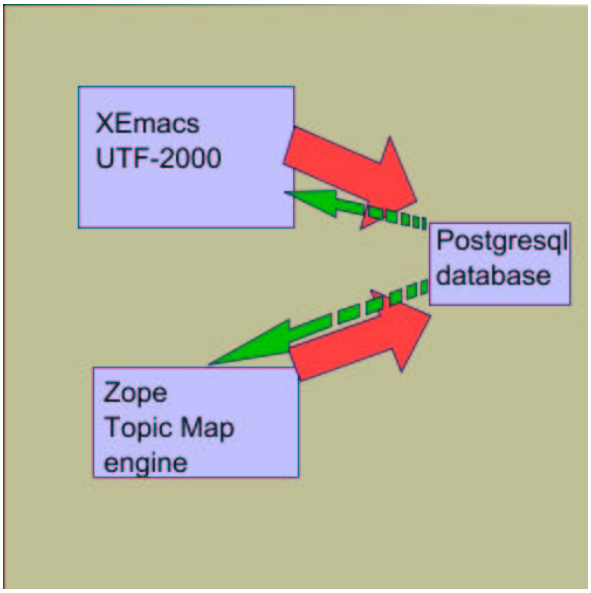

図 8: Communication between XEmacs UTF-2000, Zope and a relational database server

現在の所、Topic Maps 処理一般および XEmacs UTF-2000 の外部文字データベース・バックエンドとして用いる場合に必要となる複雑な問い合わせを効率良く処理可能な形に、関係データベース・モデルに基づく表へ Topic Map の基礎構造を写像する方法を開発している所である。

6 参加企業及び機関

なし。

7 参考文献

[1] International Organization for Standardization (ISO). Information processing — Text and office systems — Standard Generalized Markup Language (SGML), 1986. ISO 8879:1986.

[2] International Organization for Standardization (ISO). Information processing — Text and office systems — Hypermedia/Time-based Structuring Language (HyTime), 1997. ISO 10744:1997.

[3] International Organization for Standardization (ISO). Information technology — SGML Applications — Topic Maps, January 2000. ISO/IEC 13250:2000.

[4] International Organization for Standardization (ISO). Information technology — Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane (BMP), March 2000. ISO/IEC 10646-1:2000.

[5] International Organization for Standardization (ISO). Information technology — Universal Multiple-Octet Coded Character Set (UCS) – Part 2: Supplementary Planes, November 2001. ISO/IEC 10646-2:2001.

[6] Bil Lewis, Dan LaLiberte, and Richard Stallman. GNU Emacs Lisp Reference Manual. Free Software Foundation, 2.5 edition, May. for Emacs Version 20.3.

[7] Richard M. Stallman et al. GNU Emacs version 21.2. ftp://ftp.gnu.org/gnu/emacs-21.2.tar.gz, March 2002.

[8] The Unicode Consortium. The Unicode Standard, Version 3.0, February 2000.

[9] XEmacs. http://www.xemacs.org/.

[10] Zope. http://www.zope.org/.