2004－CH－62　(8)
2004／5／28

文字素性に基づく文字処理

守岡 知彦　　師 茂樹
京都大学人文科学研究所　　花園大学

概 要

文字や文字の性質を文字符号に制約されることなく自由に表現･処理するための環境として開発を進めている CHISE (CHaracter Information Service Environment) システムと、そこで用いられている文字処理技術について概説する。

1　はじめに

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）を情報の発信者と受信者の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・政治的問題があり、根本的な解決にはいたっていない。

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といえる。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 [1] / XML [6] 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。

我々はこのような考えを実証するために CHISE (CHaracter Information Service Environment) Project をはじめた。その主要な目的は、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・蓄積した文字オントロジを実現するとともに、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、文字素性に基づく文字処理技術を提案し、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。

2　Chaon モデル

符号化文字方式に基づかずに文字を表現するために、我々は「Chaon モデル」と呼ぶ文字表現モデルを提案している。これは、指し示したい文字の性質を列挙することで文字を表現しようというものである。ここで、この「文字の性質」のことを『文字素性』(character feature) と呼ぶ。

素性（feature）の概念は、ローマン・ヤーコブソンの音韻論にある弁別素性（distinctive feature）の概念から援用している。その理由は、「文字の性質」が文字の何らかの実体に従属しているもの（例えば「属性（attribute）」や「プロパティ」などの用語で表されるもの）ではなく、性質の集合そのものが文字である（性質がなくなれば、文字もなくなる）ということを明示したいためである。抽象的な文字を措定し、それに一意のコードポイントを与え属性を付する Unicode に代表される文字コードの考え方（[10] 参照）とはまった

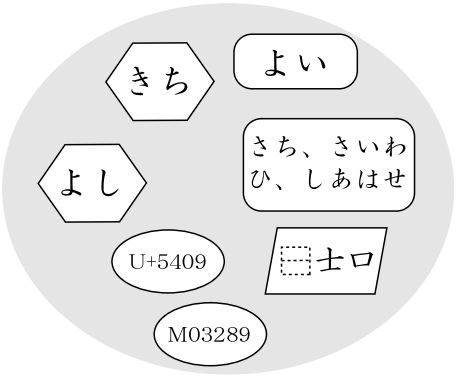

図 1: Chaon モデル

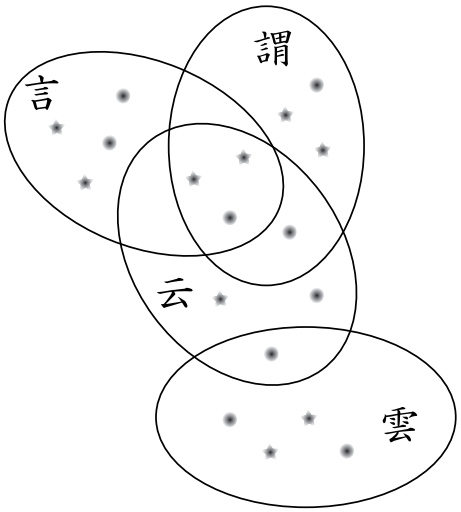

図 2: 文字オブジェクト間の比較

く異なる。

したがって、Chaon モデルにおいては、文字素性の集合によって表現された文字を「文字オブジェクト」とよぶが、上記の理由からこの「オブジェクト」は、一般的なクラスベースのオブジェクト指向における「オブジェクト」とは異なる。

文字素性には、字形・発音・意味や、既存の文字コードのコードポイントなど、文字に関するさまざまな要素が考えられる。図 1 は、「吉」という文字を Chaon モデルで表現したイメージである。

ちなみに、先に Chaon モデルが文字コードのモデルとまったく異なると述べたが、ある特定の文字コードに素性を限定すれば、文字コードによる文字処理と同じになる。したがって、文字コードモデルは Chaon モデルの部分集合と言うこともできる。

文字オブジェクト間の比較は、図 2 のような集合演算になる。この図では、「言」「云」「謂」の字が、字形は異なるものの意味や発音などの素性で重なる部分があること、また「云」と「雲」が、中国においては発音や意味などの素性が共通することなどを示している。

このような比較方法に類似する方法としては、国語学者の野村雅昭氏が提案する漢字の比較モデル（[8]）を挙げることができる。このモデルにおいては、文字の構成要素である形・音・義を、それぞれ字体素・音素・意義素とよび、文字を構成するという点で同等の資格を持つとする。そして文字の比較においては、各要素ごとに比較することで、同字・別字以上の複雑な関係を記述可能にしようとするものである。Chaon モデルは野村氏のモデルとは独立して考案されたものであるが、国語学上のモデルと共通性を持っている点は、モデルの妥当性を考える上で考慮されるべきではなかろうか。

なお、図 2 のような集合演算は、後述する Perl/CHISE においてすでに実装されている。しかしこれは単純化したものであって、実際の比較においてはコンテクストを考慮した集合演算が行われなければなるまい。すなわち、上にあげた「云」と「雲」の例は、日本においてはそのままでは適用できないので、素性の扱いに制限を加えなければならないだろう。

3 CHISE システムの概要

Chaon モデルに基づく文字処理は、文字をコードではなく文字素性の集合として扱い、文字素性によって操作するものとなる。このことは、Chaon モデルに基づく文字処理系が文字オントロジを操作するためのある種のデータベース・システムとなることを意味する。

このため、CHISE Project では、CHISE 環境で文字情報を共有するための文字データベース処理系およびそのコンテンツ、そして、文字データベースを参照して文字処理をアプリケーションの開発を行っている。

具体的に、

libchise 文字データベース操作のための基本機能を提供するためのライブラリ

XEmacs CHISE XEmacs [7] に基づく Chaon 実装

（Emacs Lisp 処理系および対話型編集環境）

Ruby/CHISE Ruby [5] に基づく Chaon 実装（スクリプト言語）

Perl/CHISE Perl に基づく Chaon 実装（スクリプト言語）

Ω/CHISE 多言語 *TeX* 処理系 *Omega* [4] に基づく Chaon 実装（組版システム）

Kage 漢字グリフ合成システム [9]

といったシステムの開発が行われている。

XEmacs CHISE, Ruby/CHISE, Perl/CHISE は CHISE 環境の文字データベースを読み書きすることができる。また、Ω/CHISE は参照のみを行い処理を行う。

4 文字素性の操作

XEmacs CHISE, Ruby/CHISE, Perl/CHISE は Chaon モデルに基づく文字操作機能を提供している。ここでは、一番実装が進んでいる XEmacs CHISE を例にその概要を示す。

XEmacs CHISE では、Chaon モデルに基づき、文字（または、文字の集合を表す『文字状況』）を文字素性の集合によって表現している。文字素性は素性名と値の対であり、キー部が素性名を表すシンボルである cons 対で表現することができる。また、文字素性の集合は連想リストで表現することができる。この連想リストはこれによって表現される文字の性質を表現したものと考えることができる。そこで、このような連想リストを『文字指定 (char-spec)』と呼ぶことにする。

文字指定は概念的には文字であるが Lisp 的にはリストの一種であり、そのままでは XEmacs における文字型オブジェクトとは見做されないことになる。このため XEmacs CHISE では、文字指定の集合を文字データベースに格納するとともに、そこでの文字指定への参照である “char-id” を文字型オブジェクトの内部表現とし、文字データベースを介して文字素性ないしはその集合である文字指定と文字型オブジェクトを結びつけている。

結局、概念的には CHISE 文字データベースは連想リスト（文字指定）の配列と考えることができ、そのインデックスが char-id である。char-id は従来型の文字の内部コードのようなものを必要とする時にその代用品として用いることができ、それによって従来型文字処理プログラムを最小限の労力で Chaon 実装化することができる。

以下に XEmacs CHISE が提供する文字関連の機能を示す：

関数 define-char (*char-spec*)

文字素性の集合 *char-spec* で表現される文字オブジェクトを定義し、その文字オブジェクトを返す。

[例]

```
(define-char
  '((name             . "CJK RADICAL SECOND TWO")
    (general-category symbol other)
    (bidi-category    . "ON")
    (mirrored         . nil)
    (total-strokes    . 1)
    (<-radical
     ((=ucs           . #x4E5A)
      ))
    (=big5-cdp        . #x8C5D)
    (=ucs             . #x2E83)
    ))
```

関数 get-char-attribute (*character feature* ***&optional*** *default-value*)

文字オブジェクト *character* の素性 *feature* の値を返す。

もし、値が定義されていない場合、*default-value* を返す。なお、*default-value* の既定値は nil である。

[例]

```
(get-char-attribute ?あ 'name)
→ "HIRAGANA LETTER A"
```

関数 put-char-attribute (*character feature value*)

文字オブジェクト *character* の素性 *feature* の値を *value* に設定する。

[例]

```
(get-char-attribute ?あ 'foo)
→ nil
(put-char-attribute ?あ 'foo 1)
→ 1
(get-char attribute ?あ 'foo)
→ 1
```

関数 remove-char-attribute (*character feature*)

文字オブジェクト *character* の素性 *feature* を削除する。

関数 find-char (*char-spec*)

文字素性の集合 *char-spec* で表現される文字オブジェクトを探索し、見つかった場合はそれを返す。見つからなかった場合は nil を返す。

関数 map-char-attribute (*function feature*)

文字素性名 *feature* を持つ各文字に対し、関数 *function* を適用する。

この関数は2引数の関数であり、この関数が呼ばれる時、第1引数に文字が渡され、第2引数に文字素性値が渡される。また、この関数が非-nil 値を返すと、関数 map-char-attribute の実行はその時点で停止される。

関数 char-attribute-alist (*character*)

文字 *character* に対応する文字指定を返す。

関数 char-attribute-list ()

現在までに使われた文字素性名の一覧を返す。

XEmacs CHISE は CHISE 環境で共有される文字データベースの他にプロセス内部のオンメモリ・データベースを持っている。オンメモリ・データベースに存在しない文字データは最初に必要となった時に共有文字データベースからオンメモリ・データベースに読み込まれる。また、オンメモリ・データベースに書き込まれた情報はそのままでは共有文字データベースに書き込まれず、そのプロセスの外部からは参照できないし、プロセスの終了とともに消滅することになる。そのため、オンメモリ・データベースを永続化するための機能が用意されている：

関数 save-char-attribute-table (*feature*)

各文字の文字素性 *feature* の値を CHISE 文字データベースに書き出す。

また、プロセス内部の文字データベースにおける情報を破棄して、CHISE 文字データベースの情報を読み直せるようにするための機能も用意されている：

関数 reset-char-attribute-table (*feature*)

プロセス内部の文字データベースから各文字の文字素性 *feature* の値を消去し、CHISE 文字データベースの情報を読み直せるようにする。

関数 reset-charset-mapping-table (*coded-charset*)

プロセス内部にある *coded-charset* の decoding-table を破棄し、CHISE 文字データベースの情報を読み直せるようにする。

また、文字素性に関する情報を一気に読み込むための機能も用意されている：

関数 load-char-attribute-table (*feature*)

CHISE 文字データベースから各文字の文字素性 *feature* の値を読み込む。

この他、文字素性を陽に登録するための機能も存在する：

関数 mount-char-attribute-table (*feature*)

CHISE 文字データベースから文字素性 *feature* を読み込めるようにする。

なお、Ruby/CHISE や Perl/CHISE にはオンメモリ・データベースは存在せず、put-char-attribute に相当する関数で書き込んだ文字データはすぐに共有文字データベースに書き込まれる。そのため、オンメモリ・データベースと共有文字データベースの同期をとるための一連の関数は存在しない。

5 文字素性

5.1 操作に基づく分類

Chaon モデルに基づく文字処理システムを実際に運用する上で重要となるものは、どのようなものを文字素性として捉えるとともにそれらをどのように表現するかである。文字の性質は無数に考えることができるが、我々は文字に対する操作に対応するものとして文字素性を捉えるのが妥当であると考えている。

この観点に立ち、現在、我々は文字素性を

1. （辞書記述的な）文字の性質
2. 文字の ID 等への写像

3. 文字間の関係

の 3 種に分類している。例えば、部首、画数、音価などは 1. に属し、UCS [2] での符号位置のようなものは 2. に属する。また、異体字関係のようなものは 3. に属する。

2. に属する情報は文字符号に関する処理に用いられる。文字符号に関する処理には符号化と復号化の双方の処理が必要であり、文字の 2. に属する文字素性の値を取り出すことは符号化に相当する。文字符号に関する処理は通常高速性が要求されるので、実装上、復号化のために必要となる逆引用データベースも用意している。[1]

また、異体字変換（例えば、伝統的字体の漢字を日本の常用漢字や中国の簡体字に変換するような）処理には 3. に属する情報が用いられる。

5.2 複雑な情報の記述

汎用的な文字データベースを作る場合、用途や立場・学説などによって、文字素性に値に複数の選択肢を設けたくなる場合がある。この場合、しばしば、単純に複数の値が記述できるだけでなく、それらの値の出典情報などのメタデータも付加したい場合がある。こうした場合、文字素性の値か名前のどちらかを構造化する必要がある。

そこで、CHISE では値を構造化するための形式として『文字参照 (char-ref)』と呼ぶ形式を定義している。これは S 式における属性リストの一種で、属性名がメタデータの種類を表し属性値はその値を表す。但し、属性名が `:char` の場合の属性値はメタデータが付加される文字を表す。なお、現在、意味が定義されている属性名としては `:sources` があり、この値は出典情報を表す。

CHISE では文字素性値ではなく文字素性名を構造化するための形式も定義している。これは対象となる文字素性の名前（文字素性基底名）に、『ドメイン識別子』と呼ぶ値を選択するための識別子を付けた文字列生成しそれを名前（文字素性具象名）として用いたり、同様にメタデータ識別子を付けた文字列を生成しそれを名前（文字素性メタデータ名）として用いる方法である。この名前は次のような規則で生成される：

[1] この特別扱いのために 1. と 2. を区別しているともいえるが、あまり本質的な差異ではないのかも知れない。

文字素性具象名
:= 文字素性基底名 @ ドメイン識別子
| 文字素性具象名 / ドメイン識別子

文字素性メタデータ名
:= 文字素性具象名 * メタデータ識別子

例えば、総画数を表す文字素性名を `total-strokes` とし、ドメイン識別子として `ucs` を用いる時、文字素性具象名は `total-strokes@ucs` となる。また、出典情報を表すメタデータ識別子を `sources` とする時、`total-strokes@ucs` の出典情報は `total-strokes@ucs*sources` で表される。

部首と部首内画数のように異なる種類の文字素性の値が対応関係を持っている場合、ドメイン識別子を用いてその対応関係を表すことができる。例えば、部首を `ideographic-radical`, 部首内画数を `ideographic-strokes` で表す時、

```
ideographic-radical@ucs
ideographic-strokes@ucs
```

の両者は対応する。

値を構造化する手法と名前を構造化する手法を比べた場合、前者はドメイン識別子を必要としないという利点を持っているものの、値が構造データとなるので C のような単純な記憶管理機構しかない環境では不便である。また、高速化を要するような単純な処理の場合、大抵、複数の値やメタデータを必要としないといえる。また、Chaon モデル的には文字が文字素性の単純な集合になっている方が自然であり便利であるが、値を構造化すると複数の値同士の集合演算を要し、処理が複雑になる。このようなことを鑑み、現在の CHISE では共有文字データベース内では原則として値ではなく名前を構造化する方針を採っている。

5.3 文字定義の継承

多数の異体字を含む大規模な文字データベースを構成する場合、異体字関係にある文字は多くの文字素性を共有していることが多い。そのため、CHISE では文字定義の継承を導入している。

6 CHISE 文字データベース

Chaon モデルに基づく文字処理システムを活用するためには文字知識のデータベース化が不可欠であり、我々はこうした文字データベース・コンテンツの開発にも力を入れている。現在、CHISE 環境用に提供されている文字データベースには

1. XEmacs CHISE 附属のデータベース
2. CHISE 漢字構造情報データベース (CHISE-IDS)

の２つがある。

前者は XEmacs CHISE に附属する文字データベースで、XEmacs CHISE の define-char 関数を用いた Emacs Lisp プログラム（define-char 形式）で表現されている。

後者は ISO/IEC 10646-1:2000 [2] で定義された IDS (Ideographic Description Sequence) を用いて漢字構造情報を表現したデータベースと XEmacs CHISE 用関連プログラムからなるパッケージである。

この漢字構造情報というのは漢字の部品の組合せ構造に関する情報のことである。ご存知のように、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が、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、CHISE 文字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計画した。そして、2001 年度には未踏ソフトウェア創造事業の予算を得て、漢字の部品の組合せ構造を表現した CHISE 漢字構造情報データベース (CHISE-IDS) の開発を始めた。現在、ISO/IEC 10646-1:2000 [2] 基本統合漢字（Unicode の例示字形）, 同 統合漢字拡張 A, および ISO/IEC 10646-2 [3] 統合漢字拡張 B に対する入力を一応完了しており、これを元に JIS X 0208:1990（の例示字形）、大漢和文字などに対する編集作業も行っている。

CHISE-IDS の開発以前から存在する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。また、この他、日本でも「今昔文字鏡」がある。CDP のデータベースと CBETA の外字データベースはそれぞれ独自形式を採っており、そのままでは IDS 形式に変換することはできない。また、今昔文字鏡のデータは一般には公開されていない。「今昔文字鏡」は専有的 (proprietary) ソフトウェアであり GPL によって配布されている XEmacs CHISE で利用することはできない。一方、CDP のデータベースと CBETA の外字データベースは 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。

XEmacs CHISE 附属のデータベースと CHISE-IDS パッケージは今のところ別々に配布されているが、XEmacs CHISE が利用可能な環境では、CHISE-IDS パッケージ附属のインストーラーを使って、CHISE-IDS データベースを XEmacs CHISE 附属のデータベースに統合することができる。また、CHISE-IDS パッケージは漢字構造情報を XEmacs CHISE で利用するためのプログラムを含んでいる。これは、現在の所、

ids.el IDS 形式の構文解析器など

ids-read.el CHISE-IDS データベースを XEmacs CHISE に読み込むプログラム

ids-dump.el XEmacs CHISE 内部で用いている ideographic-structure 形式のデータを CHISE-IDS データベース形式で書き出すプログラム

ids-util.el 漢字構造情報を Unicode 例示字体風や大漢和辞典風に変換するプログラム

ids-find.el 部品によって漢字を検索するためのプログラム

からなる。

ids-find.el は現在 ids-find-chars-including-components（別名 ideographic-structure-search-chars）と ids-find-chars-covered-by-components という２つの検索用コマンドを提供している。前者は M-x ideographic-structure-search-chars [CR] 部品列 [CR] で指定された部品列の各部品を少なくとも１個は含んでいる漢字の一覧を表示するコマンドである（図 3)。後者は M-x ids-find-chars-covered-by-components [CR] 部品列 [CR] で指定された部品列中の任意の部品を０回以上用いて全体として２個以上の部品から構成される漢字の一覧を表示するコマンドである（図 4)。

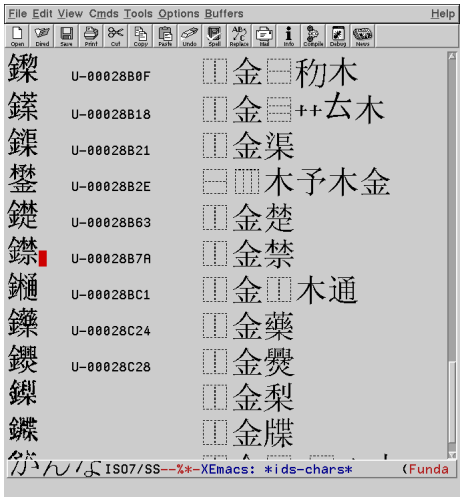

図 3: M-x ideographic-structure-search-chars [CR] 木 金 [CR] の出力結果

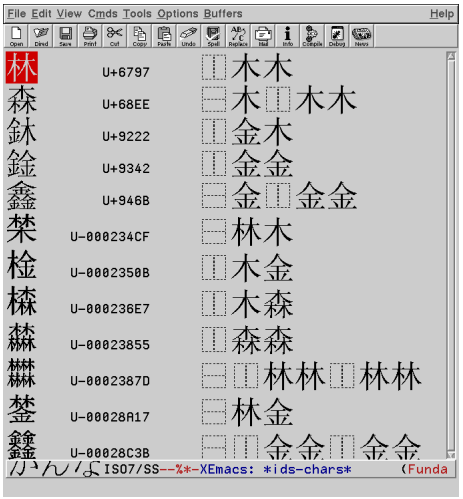

図 4: M-x ids-find-chars-covered-by-components [CR] 木金 [CR] の出力結果

7 おわりに

文字オントロジに基づく文字処理環境を開発している CHISE Project での文字処理の概要について述べた。CHISE Project は現在の豊富な計算資源を背景に文字に関するさまざまな知識をデータベース化し、それを参照することによって文字やテキストを処理するシステムを開発している。我々はこのシステムのために、従来の符号化文字モデルに代わる文字表現モデルとして、「Chaon モデル」と呼ぶ文字表現モデルを提案している。これは、指し示したい文字の性質を列挙することで文字を表現しようというものである。ここで、この「文字の性質」のことを『文字素性』(character feature) と呼ぶ。この手法は表現したい文字が満たすべき性質を十分に表現可能にしており、文字に関する知識を記述する上で必要とされる表現力を兼ね備えており、また、実装も容易である。

我々は文字オントロジに基づく文字処理の可能性を現実のものとし、文字やテキストを多面的に操作する文書処理環境の実現するために、文書編集系 XEmacs CHISE, スクリプト言語 Ruby/CHISE, Perl/CHISE, 多言語組版系 *Omega*/CHISE を開発し、Chaon モデルの実現可能性や有効性を実証している。

しかしながら、このようなシステムの有効性は文字に関するさまざまなデータベース・コンテンツの充実度に依存しており、画面表示や印刷においてはグリフデータの拡充が重要な課題となっている。また、Chaon モデルを活かしたアプリケーションの充実も重要である。よって、プログラム・データ双方のさらなる整備が必要であり、多言語化技術、国際化技術、データベース、分散化技術など各方面のハッカーや、言語学、文字学、文献学など各種言語や字書やテキストなどを扱う人文科学者など、多方面に渡る研究者の幅広い協力が不可欠である。

こうしたことを鑑み、CHISE プロジェクトはオープン・ソースで開発されており、その情報は

- http://cvs.m17n.org/chise/
- http://kanji.zinbun.kyoto-u.ac.jp/projects/chise/
- http://mousai.as.wakwak.ne.jp/projects/chise/

で公開されている。これらの WWW 頁群を含め、各種成果物は CVS で管理されており、最新の開発状況を知ることができる。日本語用と英語用の 2 つのメーリングリストも用意されており、参加方法は上述の WWW 頁で説明されている。CHISE プロジェクトに興味を持たれた方は、是非、お気軽に御参加願いたい。

謝辞

本論文で述べた CHISE プロジェクトの研究の一部は 2000 年度に旧通商産業省工業技術院電子技術総合研究所（現 独立行政法人 産業技術総合研究所）からの受託研究として行われ、2001 年度には情報処理振興事業協会の「未踏ソフトウェア創造事業」の助成を受けた。

また、忙しい中 CHISE プロジェクトに主要メンバーとして御参加頂いた Christian Wittern 氏、江渡浩一郎氏、上地宏一氏、鈴木泰博氏、苫米地等流氏、藤原義久氏、宮崎泉氏に感謝する。また、2001 年度に未踏ソフトウェア創造事業のプロジェクトマネージャーとして、その後も、プロジェクト遂行において貴重なご助言と多大な御助力を頂いた g 新部裕氏に感謝する。

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 technology — Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane (BMP)*, March 2000. ISO/IEC 10646-1:2000.

[3] 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.

[4] The omega typesetting and document processing system. http://omega.cse.unsw.edu.au:8080/.

[5] The object-oriented scripting language Ruby. http://www.ruby-lang.org/.

[6] The World Wide Web Consortium (W3C). *Extensible Markup Language (XML) 1.0 (Second Edition)*, October 2000. http://www.w3c.org/TR/2000/REC-xml-20001006.

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

[8] 野村雅昭. 同字と別字のあいだ. 日本語学, Vol. 3, No. 3, 1984.

[9] 上地宏一. 漢字フォント自動生成サーバ “影 KAGE” の構築 — 文字コードの枠組みを越える次世代漢字処理の提案 —. 漢字文献情報処理研究, Vol. 3, pp. 143–147, 2002.

[10] 師茂樹. Unicode の *character* 概念に関する一考察. 東洋学へのコンピューター利用 第 14 回研究セミナー, 京都大学学術情報メディアセンター 第 71 回研究セミナー, pp. 3–8, Mar 2004.