ウェブ方式の古書目録DB用データ入力システムの開発

——外字問題を中心に——

北村啓子

要　旨　古書目録DBのようにJIS外字を含む大規模DBを分散環境で多数の人の共同作業で構築することを想定し、ネットワーク環境でウェブ（Javaサーブレット）技術を使ったデータ入力システム開発の例と、全文検索エンジンを使った目録検索システム開発の例を紹介する。それぞれのシステムについて、UNICODEを使ったJIS外字の入出力の実現方法、その場合の問題点と解決方法について説明する。

1．はじめに

Web技術の発展により分散作業型のDB構築システムが可能になり、またWebブラウザでのUNICODEサポートにより容易に多数の日本語を扱えるようになった（JIS第1第2水準 6,355文字 UNICODE 20,902文字）。JIS第2水準以上の日本語を扱うために長年苦労してきた国文学などの分野には朗報である。しかし実際に第2水準以上の日本語DBのデータ入力をネットワーク環境で行うシステムを構築しようとすると、様々な問題にぶつかる。UNICODEサポートに関してはWebブラウザ先行になり、データを表示するには問題ないが、DBへのデータ入力をしようとするとDB内部コードとの変換が発生したり、また内部処理するプログラミング言語のUNICODE対応が遅れていたり、文字コード変換の規則が何種類もあったり、文字コードの組み合わせによっては矛盾が発生したりするのが現状である。

本稿では、Javaサーブレット技術を使った古書目録DBへのデータ入力システム開発の例、ならびに全文検索エンジンを使った目録検索システム開発の例を紹介し、日本語文字コードに関する問題と解決方法につい述べる。

2．目録DBへのデータ入力システムの開発例

2-1. 経緯

平成9年から当館の目録データベースは従来のメインフレーム型リレーショナルDB（RDB）からワークステーション型オブジェクト指向DB（OODB）に再設計、データ移植が行われた[1)]。引き続き複数DBの統合化を機に、典拠参照型のローカル目録への再構築が行われてきた。これは独立典拠を参照する形

で典拠コントロールされた新しいローカル目録を容易に効率よく作成することを可能にするものである[2)]。

このOODBに移植された目録データベースにデータ入力を行うシステムが必要であり、「サーブレット」技術を利用して開発することになった。サーブレットとはWeb技術とJava（ジャバ：プログラミング言語）の長所を統合した仕掛けであり、既に多くの商用システムで使われている実績を持つ。サーブレットは従来のWebアプリケーション技術に比べ少ないリソース（計算機資源）で高速な処理能力に富み、マルチユーザ・マルチタスク処理に強く、多数のユーザが同時にDBアクセスするような処理やトランザクション処理に適している。さらにプラットフォーム非依存性による高いポータビリティ（多くの機種で稼働可能）、高い並列処理機能などJavaのメリットをそのまま活かすことができる。

2-2. データ入力システムの考え方

設計するに当たって次の3点を大きな目標に据えた。

a. 大規模目録DBを分散環境で多数の人の共同作業という形態で構築できること。目録データ入力システムは図書館内でクローズしがちであるが、古書の場合本自身が全国、世界中に分散して存在し、現地で実物を見ながら、または所蔵図書館の司書などが直接データ作成に参加する共同作業型は効率よく高品質なDBを構築する有効な方法として考えられる。システム的には海外も含みインターネットにつながっていれば遠隔地からでも参加可能とする。

b. UNICODE（Webブラウザ　UTF-8）を採用することにより当館JIS外字中、UNICODEに定義されている文字を入出力可能にすること。さらに残る

UNICODE外字について、少数になれば実現可能性の出てくる入出力方法の開発に取り組む。

c. 典拠を中心に据え、それとのリンクを張りながら典拠参照型の個別の新しい目録DBを構築できるシステムを提供すること。

その他、設計上で重要視した点をリストアップしておく。

・古書現物を調べながらデータ入力が可能
・PCの種類を問わずWebブラウザがあれば動く
・同時に多数人のデータ入力を処理でき、十分な速さのレスポンスが得られる
・典拠検索、著者著作同定、リンク張り（典拠コントロール）の一連の作業が容易に可能
・典拠も更新され成長していくのでリアルタイムに最新の典拠を参照できる
・典拠更新時に参照している目録データへの影響をチェックし整合性を保てる
・マルチウィンドウ使用、わかりやすい画面操作
・インターネットで慣れた画面操作（フォームを使用）

2-3. 外字対策

古書目録のDBはJIS外字の問題を避けては通れない。メインフレーム上では固有の外字コードを定義、フォントを作成し、直端末と印刷でフォントを使えるよう維持されてきた。ワークステーションへの移植の際、EUCへのコード変換においてXMLで固有定義した実態参照（&Kxxxx;）として当館外字コードは残された。幸い外字2,517文字中1,857文字がJIS X 0212（補助漢字）で定義されており、補助漢字フォントはUNIX X Window用が作成されていたので、当館外字－補助漢字マッピングテーブルを介してX上での表示、PostScriptによる印字は可能であった。しかし、それ以外の環境では使えなかった。

Webが普及し、UNICODEの正式サポートに伴いWebアプリケーションでの外字を含むデータ作成の可能性が出てきた。Web技術は機種、言語、ローカルコードなどユーザ環境などを問わない汎用性の高い共通プラットフォームを提供している分、PCローカルコード→Web Server→サーブレット処理（アプリケーションプログラム内部コード）→Web Server→PCローカルコードと何度も文字コード変換が行われ、途中で不整合が起きるケースもある。

Javaは内部コードとしてUNICODEを使う。データ入力システムをJavaで開発するにあたりDB自身をUNICODEに変換することが技術的には望まれたが諸事情によりEUCのまま維持せざるを得なかった。従って、データ入力（Webブラウザ：SJIS）→Javaサーブレット処理（UNICODE）→DB（EUC）→Javaサーブレット処理（UNICODE）→Webブラウザ（SJIS）の文字コード変換が発生し、この間において文字コードの問題があることがわかった。

a. Webブラウザの使うローカルコード“Shift-JIS”に対してUNICODEへ変換するコンバータはSJIS MS932 CP943 CP943C MacTECなど複数存在し、一部の文字（～‖－¢£¬）が異なるコードにマッピングされていて、異なる文字または未定義文字に変換されてしまう。Windows NT/2000/XPでフォント定義されているメーカ定義文字（機種依存文字）848文字がUNICODEへマッピングされているのはMS932 CP943 CP943Cであり、それ以外のコンバータではJIS外のUNICODE文字をデータ入力しても変換されないことになる。ローマ数字大文字はJIS内外に二重定義されていたり、機種依存文字もNEC拡張／IBM拡張に重複定義されているのもあり、一本化する必要がある。

b. JavaがデータをOODB（EUC）に格納する時、EUCコンバータはJIS X 0212をサポートしておりすべての補助漢字が対応するUNICODEにマッピング

され、EUCの補助漢字領域（3バイトコード）に変換される。しかしながらJIS X 0212は“Shift-JIS”のどのコンバータもサポートしていないので不可逆変換となってしまう。

a.b.の両方を勘案し、Windowsに機種依存文字としてフォント定義されている文字はデータ入力できるのを救うため、独自のマッピングテーブルを作り毎回独自のマッピングフィルターをかけることにした。機種依存文字848文字のマッピングテーブルを作成し、“－～”の正しいマッピングとローマ数字大文字の一本化もこのマッピングテーブルに入れて解決した。Windowsにフォント定義されていないUNICODE文字については、WebブラウザがW３C参照文字（“&#xxxx;”）としてWeb Serverに送るので、そのままのコードでDBに格納することにした。独自のマッピングテーブルを持つ方法は、UNICODEサポート状況やコンバータの版など開発環境に左右されずに済み、またUNICODE外の文字について独自対策にも使える有効な策である。

その他、DB上の文字コードの問題として、メインフレーム上でJIS78のまま維持してきたためJIS78→83→90の変遷を追って文字コードの追加・変更、字形の入替えなどの変換をする必要があった（詳しい内容は[3] 第Ｊ章 文字表と対応表 pp377-384)。また既に誤ってEUC3バイトコードに変換入力された補助漢字や長年蓄積されたエラー外字コードの修復も同時に行った。

3．目録検索システム開発の例

OODBから対象目録DBの組合せ、条件を満たすデータを抜き出し、典拠リンクを展開した個々の目録データをタグでマークアップした検索用テキスト[4]を使いフルテキスト検索エンジン（Sufary：サファリ　奈良先端科学技術大学

院大学にて開発）を使った検索システムが開発された。OODBから抜き出したテキストはEUCであり、当館外字は実態参照文字で残されていた。

UNICODE（UTF-8）がWebブラウザでサポートされているのを使って当館外字を表示することにした。JIS第1第2水準6,355文字ならびに当館外字中1,857文字についてUNICODEマッピングテーブルを作成し、検索結果を毎回このフィルタを通してWebブラウザに送り、PCの持つUNICODEフォントを使って表示する方法をとった。

4. 課題と今後

（1）JIS外字の入力

UNICODEの表示は容易になったが入力は簡単ではない。Windowsでは直接コード入力する以外日本語入力フロントエンドの文字パレットで選択するのが現状である。読みデータをつけて日本語入力フロントエンドのユーザ辞書に登録するか、または使用頻度・異体字などでソートしたフォント一覧から選択できるWeb参照型のインタフェースなど入力補助手段が必要である。

（2）スタンドアロンPCでのデータ作成とDBへの自動登録

ネットワークのない場所での調査、マイクロフィルムを見ながらデータ採取など、必ずしもインターネットの整った環境でデータを直接入力するのに適さない場合のために、スタンドアロンPCで作成したデータを再入力することなくDBに登録する道筋を準備した。PCでは普段使い慣れているAccess、FileMakerなどRDB、カード型DBのソフトを使用する。データ入力システムのフォームと同じデータ定義を行い、それに従ってデータを作成する。PCソフトでCSV形式で出力したデータを使ってDBに登録するバッチ登録プログラムを作成した。バッチ登録プログラムは人間がキーボードから入力する代わり

にファイルから読み込みながらデータをサーブレットに渡すWebクライアントプログラムとして実現している。

平成15年にそれまで作り貯めていた約5千7百件のデータを実際にこのバッチ登録プログラムを使ってDBに登録を行った。

(3) 典拠更新の同期

DBの典拠データは、スタンドアロンPCでのデータ作成時に典拠同定を行う際に参照できるようPCソフトにダウンロードされている。目録作成プロセスの中で頻度は高くはないが典拠自身も追加・修正がなされていくので、DB内の典拠データとPCソフトに複製した典拠データとの同期の問題が出てくる。更新の頻度に合わせて適宜（または定期的に）ダウンロードをするなど運用上の考慮が必要である。

(4) 分散型DB

海外の図書館などとの共同作業を想定するとDB自身も大規模になるので物理的に分散させて運用することが考えられる。OODBの持つ分散型DBの機能を使うことで、例えばアメリカ、ヨーロッパ、アジアの拠点に分散型DBを配置し、どこからでも一つのDBとして使用することが可能である。

（報告が遅れたが本稿の内容は平成13－15年度開発分である。）

参考文献

1）人文科学系研究向けマルチメディア統合システムの研究、科学研究費基盤（B）(2)（09558043）研究成果報告書、平成11年3月（1999)、研究代表者：丸山勝巳

2）目録データベースの統合化による古典籍調査の総合的ネットワーク方式の開発－「著作典拠データベース（国文学研究資料館作成）の活用を基軸に－、科学研究費補助

金基盤（A）（2）（10351003）研究成果報告書、平成13年３月（2001）、研究代表者：松野陽一
3）日本語情報処理、Ken Lunde著、ソフトバンク　O'Reilly & Associates, Inc.
4）人文科学系の研究と情報流通を支援するための電子資料館システムの研究、科学研究費基盤（B）（2）（08451095）研究成果報告書、平成11年３月（1999）、研究代表者：丸山勝巳