電子図書館とSGMLデータベース　—その理想と現実—

大山 敬三
学術情報センター
〒112 東京都文京区大塚 3-29-1
Tel: (03) 3942-6950, Fax: (03) 5395-7064, E-Mail: oyama@rd.nacsis.ac.jp

概要

本稿では、まず、電子図書館の情報資源の構成において、文書画像データベースなどとの関係において全文データベースの位置づけを明らかにするとともに、データベースの構成・検索・表示の観点からSGMLデータベースの有効性と現在の問題点を示している。つぎに、現時点でSGMLデータベース作成のために実際に行われている方法について、実例を交えて述べている。また、全文データベースに対する情報検索において、文書構造の扱い、リージョン演算による検索処理、複数の異なる文書構造を持つSGMLテキストの扱い、WWWを通じて検索機能を提供するサーバの構成方法などについても、筆者の経験に基づいて述べている。

キーワード

電子図書館, 全文データベース, SGML, 情報検索, WWW, PAT

Digital Libraries and SGML Databases — An Ideal and the Reality —

Keizo OYAMA
National Center for Science Information Systems
3-29-1, Otsuka, Bunkyo-ku, Tokyo 112, JAPAN
Phone: +81 3 3942 6950, Fax: +81 3 5395 7064, E-Mail: oyama@rd.nacsis.ac.jp

Abstract

This paper first describes the role of fulltext databases in relation to image databases and so on in the structure of information resources for digital libraries, and shows the effectiveness and the current problems of SGML databases from viewpoints of configuration, retrieval and display. Then the methods which are actually carried out now to construct SGML databases are described with real examples. This paper next describes, in the context of information retrieval of fulltext databases, the way to handle document structures, retrieval processing with region operations, handling of SGML texts with different document structures, and methods to construct a server to provide retrieval functions via WWW, based on the author's experiences.

Keywords

digital library, fulltext database, SGML, information retrieval, WWW, PAT

1. はじめに

電子図書館と一概に言っても、人によってその概念は大きく異なる。しかし、電子図書館システムが最低限備えるべき機能として、文書の電子的手段による検索と配布を統合的に行えることは必須であろう。本稿ではこの二つの機能に着目し、そこにおけるSGML(Standard Generalized Markup Language)の役割を示すとともに、SGMLデータベースの構築と検索の実際について述べる。

2. 電子図書館におけるドキュメントのデータ形式

配布する文書のデータ形式には、画像、ページ記述言語（たとえばPostScriptやDVI）データ、あるいはマークアップ言語（たとえばSGMLやLaTeX）データなどがありうる。

画像やページ記述言語によるデータは文書の再現性がよく、特に紙にプリントアウトしたときに原文のレイアウトに忠実な出力が得られるが、キーワードなどを用いて直接検索するには適していない。このため、通常は検索用の二次情報を追加してデータベースを構成する。

これに対し、マークアップ言語によるデータは文書の再現性は必ずしも良くないが、全文データに対して直接検索することが可能であると同時に、取得したデータを用いて読書支援機能などの二次的な利用も容易にできる。容量があまり大きくないこともデータベースの運用上の利点である。

将来的には、執筆から印刷、流通まで一貫して電子的に文書データが作成されるようになるであろうが、現在は、まだまだ印刷物だけが最終的な出版物である場合がほとんどである。また、過去の膨大な印刷資料も無視することはできない。

このような紙に印刷された大量の文書を電子化して電子図書館のデータを整備するためには、コストと時間の制約から、印刷物をスキャナーで読み込んで画像として蓄積するという手段をとらざるを得ないことが多い。しかし、このような場合でも、上記のような理由のため、マークアップ言語によるデータを並行して作成し蓄積することが望ましい。

3. 電子図書館におけるデータベースの構成

電子図書館においては、文書を検索する高度な機能と同時に、読むための文書を従来の図書館に代わって提供する機能が必要である。ここで、「読むための」というのは、ユーザの好みと情報提供者の方針に応じて、読むのに都合の良い形で文書を提供するということである。ディスプレイ上で読む人にも紙に印刷して読む人にも読みやすいという相反する要求を満たすためには、画像あるいはページ記述言語による文書とマークアップ言語による文書を適切に使い分ける必要がある。これらを考慮すると、電子図書館のデータベースサーバは図1に示すような構成となろう。

紙に印刷された文書を電子化する場合は、コストと時間の制約のため、文書は画像として入力し、検索のために必要最低限の二次情報を人手で付加することが多い。画像から正確なマークアップテキストを自動的に生成することは非常に困難であり、画像解析とOCRを用いて自動変換する試みも行われてはいるが、誤り率を無視できるまでの精度は期待できない。そこで、このような方法で作られた電子化文書を電子図書館で扱う場合は、予め自動的に生成された誤りを含むマークアップテキスト（OCRデータなど）に対して全文検索を行い、文書の提供は画像で行うという形態が現実的である。

一方、マークアップ言語によって記述された文書データが最終的な印刷物に対応した形で利用可能な場合は、データベースの持ち方にも柔軟性が出てくる。マークアップデータから印刷イメージを再現するメカニズムが利用可能でない場合は、これとは独立に画像を入力して蓄積しておく必要があるかもしれない。しかし、マークアップデータから印刷イメージの再現が可能な場合は、ページ記述言語によるデータや画

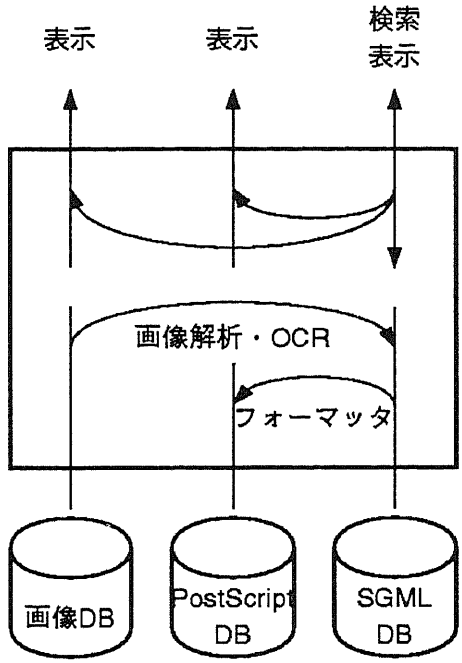


図1　電子図書館サーバの構成

像データには、予め生成して蓄積したり、要求があったときに生成して蓄積しておいたり、あるいは要求時に一時的に生成したり、といった選択が可能であり、文書へのアクセスの特性やサーバシステムの処理能力や外部記憶容量に応じて適切な方式を採用できる。

文書データにはさまざまなものがあり、電子図書館では上記のような、入力方法が異なったさまざまな電子化文書を統合的に扱える必要がある。

4. マークアップ言語

電子図書館における文書のマークアップの役割には、利用者に利用しやすい形で提供するために他のデータ形式に変換できるようにすることと、記述された文書構造を検索対象として利用できるようにすることの二つがある。

マークアップテキストが印刷の工程に組み込まれて作成されている場合は、一般にデータ形式の変換に必要な情報は含まれている。文書構造の検索への利用も、きちんと構造を定義し、それにしたがってマークアップされている場合は可能であるが、マークアップ言語の種類によってはこの定義に違反することが容易にできてしまうため、データベースの構築時にさまざまな問題が発生する可能性がある。

マークアップテキストが印刷とは別工程で作成されている場合は、必ずしも正確に他のデータ形式に変換できるだけの情報を持っているとは限らない。このような場合には配布用に画像データを蓄積しておく必要が生じ、余計なコストが発生する。文書構造の検索への利用については上の場合と同様である。

画像データからマークアップテキストを自動生成する場合は、データ形式の変換の役割は最初から考える必要がないので、検索に有効な文書構造のみを抽出してマークアップテキストを作成することになる。

上記の役割を考慮すると、マークアップ言語として以下のような条件を満たす必要がある。

(1) 論理構造のみを記述するようになっていること

- 余計な書式制御などが入っていると検索の邪魔になる

(2) 検索に必要な論理構造をすべて記述するようになっていること

- 記述が不揃いであったり不備があると見つかるべきものが見つからなくなる

(3) 構造が明確に定義されていること

- 構造が明確でないと、データベース構築時にフォーマットエラーや処理の漏れが生ずる

LaTeX や TeX でもこのような条件を満たすように文書を作成することは可能だが、著者や印刷所に強制することは難しい。例えば、著者が作成した LaTeX データを用いる場合、著者は刷り上がりの見栄えをよくするために改行やハイフネーションなどの書式制御を行うことがあるため、この条件に違反することが多い。また、階層構造などを定義することもできないため、論理的な構造上の誤りがあっても自動的に検出する手段がない。このため、実際にデータベースにロードするときにエラーになってデータベース管理者が苦労することになる。

その点、SGML では文書型定義 (Document Type Definition: DTD) を適切に作成することで (1) と (3) を満たすようにすることができるし、その結果として (2) も自然に満たされる。文書構造が DTD で明確に定義されているので、データの正当性は編集や印刷の各段階でチェックできるため、データベース管理者は安心してデータを受け入れることができる。

SGML は、いわば、著者や編集者の自由度を奪うことによりデータの整合性を高め、データベース構築と検索に都合の良い電子化テキストを作成し、文書データを管理するための手段として非常に有効であると言える。

SGML による全文データベースの検索には非常に強力なシステムが利用可能である [1]。現在の SGML にかかわる問題点は、エディタなどのオーサリングツールや対話的にテキストを表示するためのビューワに一般的なものがない点と、印刷品質に耐える自動的なフォーマッティングを行うシステムが整備されていない点である。しかし、SGML エディタの DynaText の日本語化が進められたり、ワードプロセッサソフトウェアである一太郎などで SGML のタグ付けが可能になるなど、オーサリングツールは徐々に利用可能になりつつある。また、perl や sgmls などのツールと Netscape Navigator などの HTML ビューワを組み合わせることで、SGML ビューワの代替とすることができる [2]。印刷用のフォーマッタにはこれまでは（少なくとも日本語文書では）TeX を用いることが多く、どうしても人手をかける必要があり、最終的な印刷物と SGML データとの間に不一致が生じやすかった。しかし、日本語 DTP の「OpenXPress」で SGML 文書を処理できるようになるなど、印刷用のシステムも利用可能になってくるものと思われる。SGML をとりまくシステム環境は全体的に改善の方向であり、より一層の整備が期待される。

5.SGML データベースの構築の実際

SGML によって文書を作成するためにはまず DTD を作成する必要がある。DTD には出現する可能性のあるすべての文書構造を正確に定義しておく必要があるため、一般の図書や雑誌に適用できる DTD の作成にはかなりの困難が伴う。

そこで、学会や大学などでの学術雑誌の SGML 化を容易にするために、学術情報センターでは 100 以上におよぶ学会誌や論文誌を調査し、これらの多くに共通に適用可能な汎用 DTD を作成した [3]。学術情報センターではこの DTD を用いて、センターの研究論文集である「学術情報センター紀要」を年 1 回、定期的に刊行している [4]。実際の原稿の例は

```
URL: "http://www.rd.nacsis.ac.jp/~oyama/paper/kiyo-95/paper.sgml"
```

で見ることができる。

DTD が準備できたとして、SGML テキストを実際に作成するための作業の流れを、論文誌を例にとり図2に示す。理想としては著者が最初から SGML で文書を執筆し、編集・印刷・電子出版・データベースのすべての過程でデータを共通のフォーマットで統合的に扱えるようにしたい。しかし、実際にはまだほとんどこのような形態は実現されていないようである。

学術情報センター紀要の刊行では、著者は基本的にプレインテキストの原稿を用意し、それを外注作業で SGML テキストに加工し、編集や校正を経て、TeX に変換して印刷している。これまでは、タグ付けは通常のテキストエディタを用いて行われており、これをパーザにかけてチェックと TeX への変換を行っていた。しかし、オーサリングツールが何とか利用可能になってきたので、今後は、著者に直接 SGML 原稿を執筆してもらうか、内部での編集の一貫としてタグ付けを行う方向に変わってゆくと思われる。

このようにして作成された SGML テキストは、現在、プレインテキストに再変換されて学術情報センターの Gopher サーバで提供されているほか、著者が実験的に運用している全文データベース検索システムにおいて検索可能となっている [6]。

しかし、ある程度の頻度で刊行されている雑誌で、このように編集の段階から SGML を適用している例には、日本化学会欧文誌など [6] があるが、全体からするとむしろ希である。電子的に編集・出版をしている雑誌では TeX や LaTeX を用いる例の方が多い。これらのデータを SGML に変換する手法については調査中であり、本格的に処理を行った経験は筆者にはないが、マクロないしはスタイルファイルが的確に作られており、定義されたものを適切に用いて記述されているテキストであれば、数式などを除き、主要な文書構造はほぼ自動的に SGML に変換が可能であると考えられる。

さらに、現在でも多くの雑誌は電算写植機（CTS）などの専用システムにより印刷されており、そのデータを変換してもプレインテキストが取得できる程度である。CTS データを入手できない場合も多く、このようなときは印刷物を出発点にして SGML データを作成せざるを得ない。また、すでに刊行された印刷物についても同様である。このような場合には、OCR 入力なりパンチ入力なりでプレインテキストを作成した上でタグ付けを行うか、紙の上にタグ付けをしてからパンチ入力をするかしか、現状では方法がなく、コストと時間が大量にかかる。文書画像解析により文書構造を識別した上で OCR 読みとりをすることにより、タグ付きテキストを自動作成する方法も研究されており [7]、早急に実運用に適用できるようになることを期待したい。

6. 情報検索に必要な文書構造

印刷に使うことを考えればかなり詳細なタグ付けが必要であるが、印刷物から SGML データを作成する場合などは、目的を情報検索に限定することによりタグ付けの負荷をかなり軽減することができる。特に、最終的な文書データの提供形式を画像とする場合は、サーチと表示の選択を行うに足る程度のタグ付けで済ませることもできる。

また、逆に、学術情報センター紀要のように、編集・印刷過程で SGML テキストが作成されていて、印刷も含めて必要な情報がすべてタグ付けされている場合でも、実際に情報検索のためによく使われる文書構造は主要な部分に限られる。

サーチのときに指定できることが必要な文書構造としては、標題や著者、キーワード、アブストラクトなどのいわゆる文献情報と、参考文献（各文献単位）、本文中の章や節の見出し程度である。近接演算をサポートしていない検索システムでは段落も必要になるかもしれない。これらの文書要素は検索対象としてデータベースに構造定義しておく必要がある。しかし、それより細かい部分のタグまでは定義する必要はあまりなく、検索文字列の一部として扱うだけで十分な場合がほとんどである。

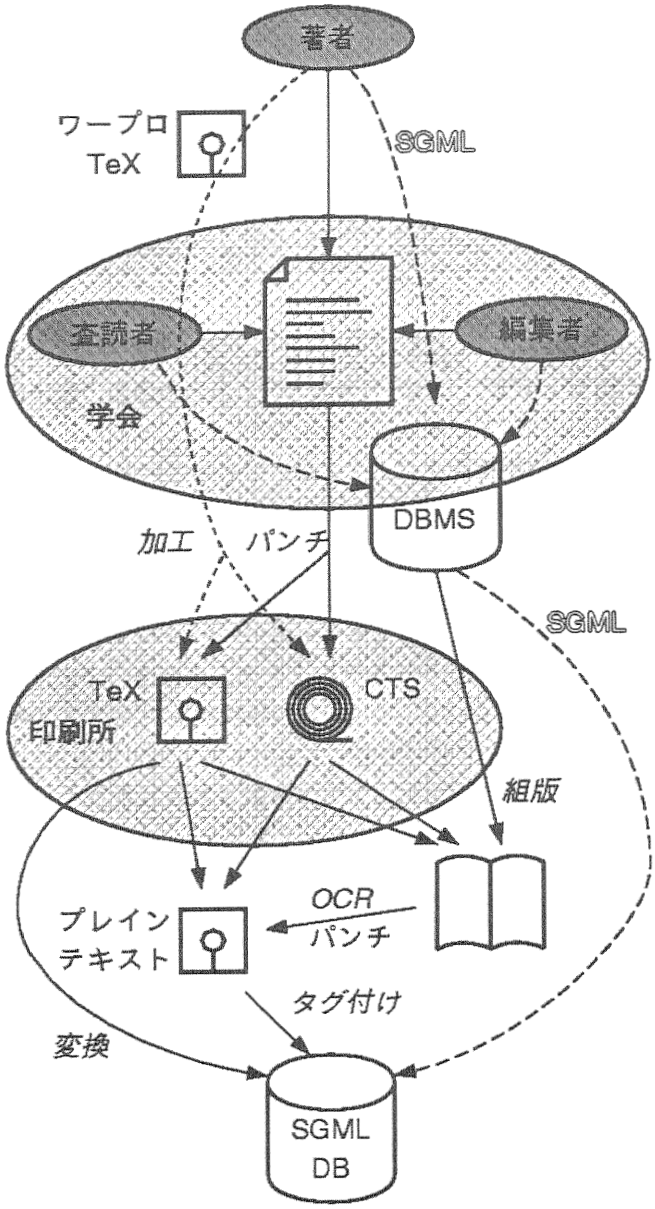


図2　SGMLデータベースの作成過程

サーチ結果の中から取り出す文書を選択するためには、文献情報、標題と章節の見出しを並べた目次、あるいは、サーチで見つかった文字列の周辺のテキストなどを表示すると有効であろう。これらはサーチの時に指定する文書要素の組み合わせで作成できるので、二重に構造定義する必要はなく、表示データを組み立てるときに動的に構成すればよい。

文書表示を、画像ではなく、SGMLテキストをフォーマットして行う場合には、詳細なタグ付けが必要である。フォーマット処理はいわば出力時のフィルタとして行われる処理であり、データベースからテキストを抽出する際には文献単位、あるいは章節単位のきわめて単純な構造が定義されているだけで十分である。後はフォーマッタがタグを見ながら処理をすることになる。

全文データベース検索システムにおいて、学術情報センター紀要データベースで実際に構造定義してサーチに利用している文書要素は、<論文>、<題名>、<著者>（一人ずつ）、<要旨>、<本文>、<章節>、章節の<見出し>、<段落>、<参考文献>（一件ずつ）の各文書要素である。文書選択用にはサーチ用と基本的に同じ文書要素を用いている。表示用としてはこれらに加えて、<文献情報>の文書要素と目次（<標題>と章節の<見出し>の組み合わせ）を用いている（いずれも実際は英文のタグ名）。

これはフリーキーワードによる検索の一般的な用途を考えて設定したものである。用途によってはこれとは異なる構造定義が必要になる。例えば、引用関係を精密に調べたければ、<参考文献>の中をさらに細かく分けて著者や題名、雑誌名などを構造定義する必要があろう。

7. リージョン演算とSGML

学術情報センターでは次期の情報検索システムのエンジンとしてカナダOpenText社のPATを採用する予定である。PATはSGMLに対応した高性能で高機能な全文データベース検索エンジンであり、文書構造をリージョンとして表現して検索処理を行っている。本章では、このリージョン演算について述べる。

リージョンとは、テキストデータ中の領域（区間）であり、通常は一つの（任意のレベルの）文書要素に対応している。これに対して、キーワードなどの文字列はその開始位置であるポイント（点）で表現される。

「著者」などの同一のタグ名を持つ文書要素は、データベースの構造定義において予めリージョンとして登録し、これらを要素とする静的リージョンセットを作成しておく。また、「目次」のように複数のタグ名の文書要素を集めたものや、特定の文書要素に含まれる見出しのように出現位置を限定したものは、通常は集合演算やリージョン演算により動的リージョンセットとして作成する。一方、キーワードなどの文字列によって検索を行うと結果は動的ポイントセットとして作成する。

開始点からなるポイントセットと終了点からなるポイントセットを組み合わせて動的リージョンセットを作成することも可能である。これにより、例えばデータベースの構造定義にない文書要素を一時的にリージョンとして扱うことができる。

集合演算では、オペランドにはポイントセットとリージョンセットをとることができ、リージョンセットは開始点だけからなるポイントセットとして演算を行う（リージョンセット同士の集合演算の場合は結果はリージョンセットとなる）。

リージョン演算には基本的に、including（含む）とwithin（含まれる）の2種類の演算がある。“A including B”は、リージョンセットAの要素の内、リージョンセットまたはポイントセットBのある要素を（区間として）含むものの集合である。また、“B within A”は、リージョンセットまたはポイントセットBの要素の内、リージョンセットAのある要素に（区間として）含まれるものの集合である。さらに、これらに否定を組み合わせたnot includingとnot withinがある。

以上からわかるように、文書要素をリージョンとして扱えば、リージョン演算によって文書要素の包含関係を表現することができる。これをDTDによる定義と組み合わせれば、SGML文書のたいていの階層構

造は扱うことができる。ただし、PAT のリージョンセットでは、集合要素のリージョン間で重なり合うことを許していないため、SGML で許されている再帰的定義だけはリージョンでうまく扱うことができない。このため、データベースにロードする前にタグ名を変更するなどして対処しておく必要がある。

実際の検索においては、いくつかの異なるタグ名を持つ文書要素を一括して検索条件中で使用したい場合（例えば<author>と<translator>など）がある。また、同じタグ名を持つ文書要素でも特定の場所に出てくるものだけを指定したい場合（例えば見出しを意味する<t>の内、図<fig>や表<tbl>のキャプションとして出てくるものだけ）もある。これらは検索のたびに集合演算やリージョン演算を組み合わせて処理することもできるが、データベース構築時に特別な処理により静的リージョンセットを作成しておくか、起動時に初期化コマンドの中で動的リージョンセットを作成しておくこともできる。こうすることで、異なる DTD を持つ文書を統合検索する場合などでもユーザに意識させずにすむ。

また、実際の表示においても、いくつかの異なるタグ名を持つ文書要素を一括して取り出したい場合（例えば目次に相当する文書要素や文献情報に相当する文書要素）がある。これらも同様にデータベース構築時か起動時にリージョンセットを作成しておくと便利である。表示する際には、出力順序を出現順に指定しておいてから文書要素を取り出せば表示のための変換は簡単な後処理だけですむし、長大な文書から一部分を取り出すような場合でも不要な部分を読み出す必要がなく効率的である。

このように、リージョン演算は単に個々の文書要素を組み合わせて検索条件を作成するだけでなく、複数の文書要素をあたかも仮想的な文書要素のように扱うためにも使えるし、出力を効率よく行うためにも効力を発揮する。

8. 複数の DTD を扱う方法

実際に文書データベースを構築しようとすると、すべての文書が同一の DTD に基づいているという状況はむしろ希であり、複数の DTD を同一のデータベースの中で扱う必要に迫られる場合の方が普通である。このような場合でも、リージョンの定義をうまく行うことにより、統合的に検索することが可能となる。

まず、検索対象として識別する必要がある文書要素（DTD ごとに異なる可能性がある）に共通のリージョン名を付ける。そして、各 DTD ごとにタグ名とリージョンの対応表を作り、静的リージョンセットを作成しておく。基本的な文書構造がほぼ同じ DTD であればこれで大体対処可能である。

しかし、DTD 間で文書構造が単純な一対一の対応になっていない場合、同一の DTD にしたがう文書ごとに 7. に述べたのと同様の方法により仮想的な文書要素に対応するリージョンセットを作成しておけばよい。これを用いれば、検索のときは DTD の違いをほとんど意識することなく統合検索を行うことができる。具体的には、データベース構築時には単純に DTD の名前と文書要素名とを組み合わせたような名前の静的リージョンセットを作っておき、起動時にリージョン演算を行ってそれぞれの仮想的な文書要素を表現する動的リージョンセットを作り、すべての DTD に対応するものの和集合を作っておくのが適当であろう。

表示の際、テキストの抽出にはこのようにして作成しておいた動的リージョンセットを用いることができるが、テキストの形式変換のためには、出力フィルターを DTD にしたがって切り換えて処理させる必要がある。

PAT にはこのように複数の文書構造からなるテキストを同時に扱うための文書グループの機能が用意されており、グループごとに独立したリージョンを作成したり、テキストとともに文書のグループ名を出力したりすることができる。この機能を用いれば上記のような複数 DTD を扱う方法が実現できる。

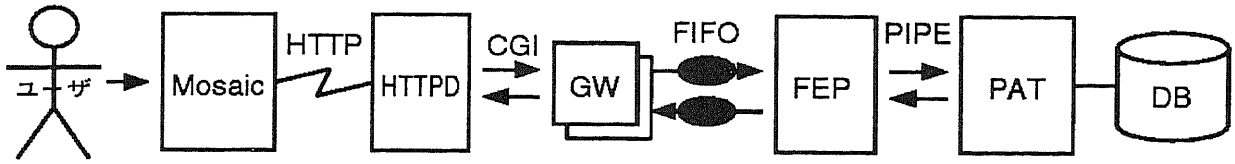


図3　WWW対応全文DB検索システムの構成

9.WWW 対応全文 DB サーバ

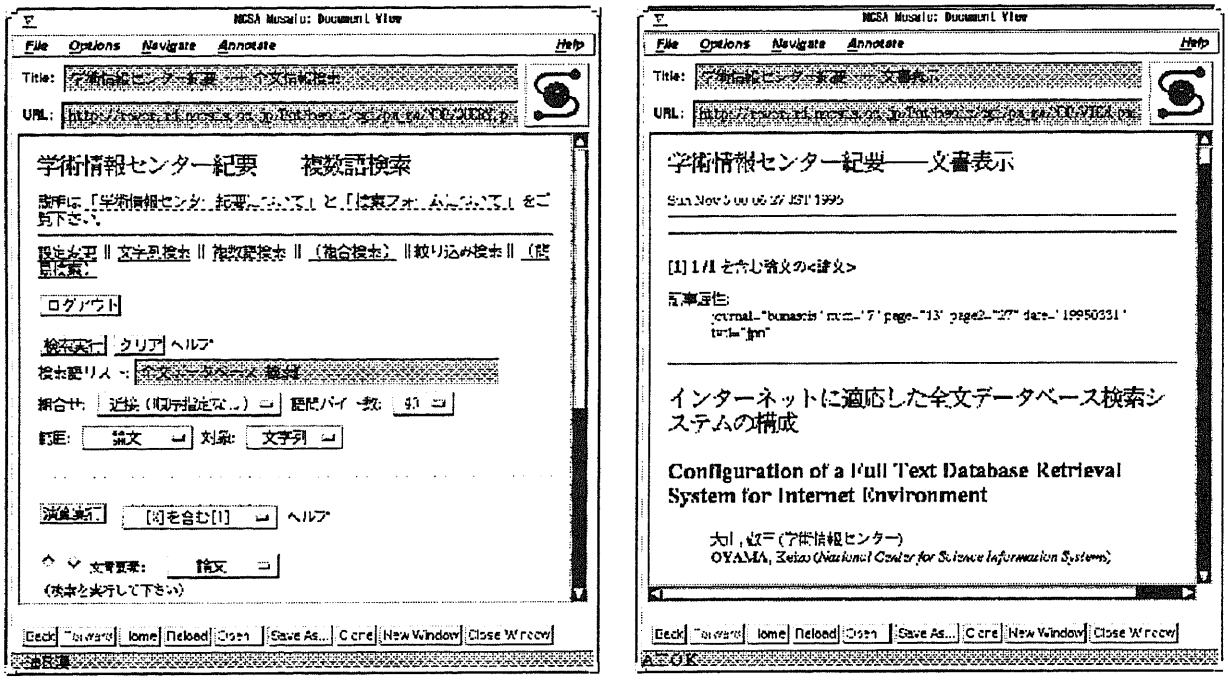
学術情報センターの次期情報検索システムでは上記の PAT を検索エンジンとして採用する予定であるが、筆者らはこのシステムのグラフィカルユーザインタフェースによるサービスとして、HTML の FORM を用いて WWW を通じて情報検索を提供するサーバを開発している。この経験を通して、全文 DB 検索機能を WWW を通じて提供する際の検討事項を述べる。

まず考慮しなければならないのは、WWW で使われている HTTP プロトコルではサーバに状態が存在しないという点である。このため、通常のシステム構成では、ユーザがそれまで行ってきた検索の履歴を保持するためには、すべての情報を FORM の中のデータに埋め込んでユーザ側に保持させる必要がある。単純な対話システムでは、検索条件を FORM に穴埋め式に書き込んでサーバに送り、サーバは検索をして結果を返して終わりとなる。結果に満足できない場合は、前とは関係なく、あらためて検索条件を書き込んで検索し直す。これならば特別の工夫をしなくとも HTTP 上に実現できる。

しかし、さまざまな条件で目的の文書を試行錯誤的に探す場合には、従来の情報検索システムのように以前の検索結果を使って新たな検索を行うというように、履歴を保持した対話システムが必要になる。この実現には、検索結果集合の代わりに過去の検索条件をすべて FORM のデータに埋め込んでおく方法と、履歴を保持する特別の仕組みをサーバ上に作り込む方法がある。幸い PAT には途中経過も含めて過去の検索結果をキャッシュしておく仕組みが備わっているので、PAT を常駐させておくようにすれば、同じ検索をやり直しても二回目はほとんど負荷にならないため、前者の方法も十分現実的である。しかし、履歴が長くなり検索条件が複雑になると、PAT のキャッシュにヒットしないものが増え、FORM 中に保持するデータ量も指数関数的に増える。このため、従来の情報検索システムのような使い方をするためには後者の方法の方が効率が良くなる。いずれにしろ、複数の検索要求にまたがって検索サーバのプロセスを継続的に走らせておく必要がある。

次に考慮すべき点は、検索のシナリオに沿った FORM の展開の仕方である。インターネットの全文索引検索サービスである Open Text Index[8] では、まず単純なキーワードだけによる検索フォームを提示し、結果を見てから文書構造や論理演算を組み合わせたより詳細な検索ができるフォームに展開するという方法をとっている。これに対し、筆者らが開発しているシステム [5] では最初からすべての検索機能を持つフォーム見せておくが、デフォルトを適切に設定しておくことで単純な検索なら簡単に行えるようにしている。参考のため、このシステムの構成を図 3 に示しておく。

さらに、さまざまな DTD やユーザの利用目的にどのように対応するかも重要な考慮点である。データベースごと、検索シナリオごとにゲートウェイプログラムを書き直すというのが最も直接的な方法であるが、データベースの提供者がプログラミングと HTML にも精通していなければならず、また開発の手間もかかる。perl などのテキスト処理ツールを使ってプログラムすれば開発の手間はある程度省けるが、プログラムコードの中に HTML テキストがちりばめられた形になり、見通しはあまり良くない。別のアプローチとして、筆者らのシステムでは HTML を生成するためにゲートウェイプログラムに外部データとして与え



(a) 検索フォームの例　　(b) 全文表示の例

図4　全文DB検索システムの表示例

るスクリプトを設計し、これを用いて図4 (a) の検索フォーム、(b) の文書表示、あるいはパラメータ設定フォームなどのすべてのHTMLデータを生成させている。

10. おわりに

本稿では、電子図書館の視点から、筆者の経験に基づき、SGMLデータベースにかかわるさまざまな話題を取り上げて述べた。SGMLを実際に使うためにはまだまだ解決しなければならない点も多いが、データベースの運用・管理をする立場から、また検索システムを開発する立場から、SGMLには捨てがたい魅力がある。今後、SGMLをめぐる環境がより改善されることを期待するとともに、筆者もそれに一役買えるよう努力を続ける所存である。

参考文献

[1] "PAT reference manual", OpenText Corp.

[2] 大山 敬三: "インターネットに適応した全文データベース検索システムの構成", 情報処理学会研究報告（情報学基礎研究会）95-FI-37, vol.95,no.45,pp.15-22(1995).

[3] "学術雑誌汎用DTD",

URL: "http://www.rd.nacsis.ac.jp/~oyama/paper/kiyo-95/paper.dtd".

[4] "学術情報センター紀要",

URL: "http://www.nacsis.ac.jp/rd/bulletin/bulletin-j.html".

[5] “全文情報検索システム”,

URL: "http://www.rd.nacsis.ac.jp/~oyama/giftir/index.html".

[6] 石塚 英弘ほか: “日本化学会欧文誌の SGML 形式全文データベースの構築・印刷そして検索”, 情報処理学会研究報告（情報学基礎研究会）93-FI-29, vol.93,no.39,pp.1-8(1993).

[7] 山岡 正輝, 岩城 修: “文書画像の SGML 文書への変換に関する一検討”, 電子情報通信学会技術研究報告（パターン認識・理解）PRU94-36, vol.94 No.241,242, pp.73-80(1994).

[8] “The Open Text Index”

URL: "http://www.opentext.com:8080/".