情　報　学　基　礎　55－5
デジタル・ドキュメント　19－5
（１９９９．７．１６）

オンラインジャーナル編集用文書処理システムの構築

大山敬三，神門典子，佐藤真一
学術情報センター研究開発部

学協会や大学などの学術機関から刊行されている学術雑誌のオンラインジャーナル化を支援するためには，多様な編集の体制や工程，投稿方式，文書ファイル形式，計算機環境などに対応できる柔軟な編集システムを構築する必要がある。著者らは，管理業務に関わる編集管理システムと，原稿自体を扱う文書処理システムをサブシステムとして構成する方式を提案して実現した。この文書処理システムでは，個別の文書ファイル形式，工程ごとの文書処理ツール，文書のバージョン，ファイル実体などの管理，および文書処理ツールの起動・監視などの機能をカプセル化し，異なる文書ファイル形式の混在や新しい文書処理ツールの導入などが容易にできるようになっている。

Development of a Document Processing System for Editing of Online Journals

Keizo OYAMA, Noriko KANDO, Shin'ichi SATOH
The National Center for Science Information Systems (NACSIS)
{oyama,kando,satoh}@rd.nacsis.ac.jp

In order to support academic institutions (e.g. academic societies and universities) to publish their scholarly journals on-line, it is necessary to develop a flexible editing system that can adapt to variety of editing organizations and flows, submission methods, document formats, and computer environments. The authors proposed and implemented a system configuration consisting of two independent subsystems, an editing management subsystem and a document processing subsystem. The document processing subsystem encapsulates the management of document file formats, processing tools, document version control, and physical location of document files. It enables editing of heterogeneous documents and easy introduction of new tools.

1. はじめに

日本において学術雑誌の多くは学協会や大学等の学術機関（以下学協会）が刊行しており，発行母体の規模が小さい場合が多いため，学術雑誌の電子化やオンライン出版が欧米に比較して著しく立ち後れている。そこで，学術情報センターでは，これらの学術雑誌のオンライン形態による出版を支援するために，「オンラインジャーナル編集・出版システム開発・構築事業」をプロジェクトとして推進し，編集プロセスを電子化して印刷原稿と同時に電子原稿を効率的に作成するための電子編集システムと，電子原稿をオンラインで出版するためのオンライン出版システムの開発を行っている。

オンライン出版システムは学術情報センターにおいて運用することを予定しており，一方，電子編集システムは利用機関に提供して運用していただくこととしている。

本稿では，このプロジェクトにおいて，電子編集システムの一部として開発した文書処理サブシステムの概要について述べることにする。なお，

本プロジェクト全体に関しては参考文献[1]を，より詳細の情報に関しては研究開発のホームページ[2]を参照されたい。

2. 全体システムの構成

本プロジェクトでは，学術雑誌のオンラインジャーナル形態による出版に必要な全ての工程，すなわち，原稿執筆から投稿，査読，編集，制作，およびオンライン出版までのプロセスの一貫したシステム化を支援することを目標としている。

システム構築の基本的方針としては，各学協会の編集工程の多様性を考え，共通機能をモジュール化してシステム開発し，共同利用すべき部分を学術情報センター，それ以外を各学協会が運用することとした。すなわち，学術情報センターは，電子ジャーナルを制作するために必要なシステムを開発して各学協会に提供するとともに，それをインターネットを通じてオンラインで購読者に提供するための基盤システムを開発・運用して学協会の利用に供する。

全体のシステムは，学協会の事務局や関係者が利用する機能を実現する電子編集システム（Electronic Editing System）と，一般の利用者にジャーナルを提供するオンライン出版システム（Online Publication System）の2つに大きく分けられる。全体システムの構成を図1に示す。システム相互間，およびシステムと外部ユーザとはインターネットによって相互接続される。

電子編集システムは，原稿や工程の管理と，原稿を編集・処理するための機能を提供する。学協会の編集担当者，編集委員，査読者，著者などが本システムの利用者である。

編集・査読体制や工程，システム環境，原稿の文書ファイル形式や処理ツールなどは学協会ごとに多様であり，これらに柔軟に適応できるようにするために，電子編集システムでは図2のようなサブシステム構成をとる。

学協会事務局内には編集管理サブシステム（Editing Management Subsystem）を運用するサーバと，編集担当者が操作するクライアントが設置され，イントラネットで相互接続される。編集担

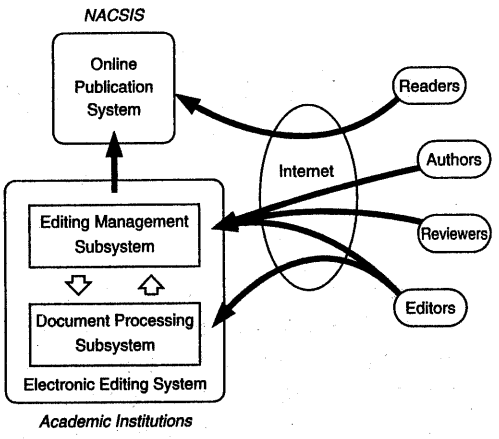

図1 全体システムの構成

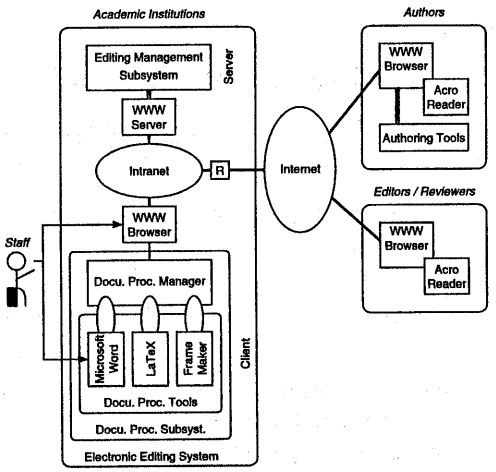

図2 電子編集システムの構成

当者はクライアント上ですべての編集業務を行う。外部から著者や編集委員，査読者などが利用する場合はインターネットを経由してアクセスする。この図では簡単のため，文書処理サブシステム（Document Processing Subsystem）はクライアント上で動作するように示しているが，実際は一部の処理はサーバ上で動作する。

編集業務の処理フローの例を図3に示す。処理フロー自体は編集管理サブシステムによって制御される。この図の中で，影を付けた処理工程にはそれぞれの内容に応じて文書処理が対応づけられており，編集管理サブシステムから文書処理サブシステムを呼び出すことにより処理を実行するよ

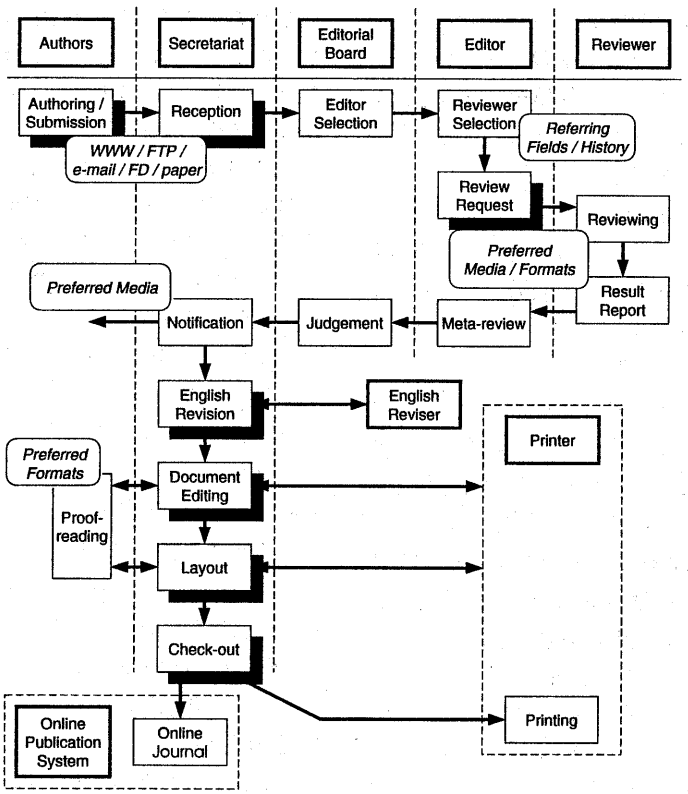

図3 編集業務処理フロー

うになっている。具体的な文書処理内容については次節で述べることにする。

3. 文書処理サブシステムの設計

文書処理サブシステムは投稿からオンライン用電子原稿を作成するまでの一連の文書処理を行うソフトウェアであり，多数のモジュールから構成される。以下にその設計方針や構成などについて述べる。

3.1 設計方針

本システムでは，原稿管理や工程管理を文書処理機能（文書ファイル形式や編集ツールなど）からできるだけ独立させることを設計の基本方針としている。また，原稿の編集やレイアウトに用いるツールはできるだけ既存のソフトウェア（例えばワープロやDTP）を利用し，必要な機能のみを開発することとした。そうすることにより，学協会ごとの多様な文書処理に対する要求への対応や，今後開発されると予想される様々な文書処理技術の導入が容易に行える。

そこで，これを実現するため，図2に示したように，文書処理マネージャ（Document Processing Manager）を中心として，多数の文書処理ツール群（Document Processing Tools）を組み込む形で文書処理サブシステムを構成することとした。

また，多様な文書ファイル形式に対して，論文管理情報（投稿票）および引用情報の抽出やオンラインでの全文検索などを一元的に行えるようにするため，サブシステム間のテキスト情報の交換用に共通文書フォーマットを開発した。これは粗い文書構造を表現するためのXML DTDとして定義されている[3]。

3.2 システム要件

(1) 文書処理内容

前節の図3の各編集工程において，以下のような文書処理を行うことが必要である。

執筆・投稿工程（Authoring / Submission）：

著者はワードプロセッサのテンプレートやスタイルファイルなどの執筆用ツールを用いて原稿を作成し，スタイルチェッカでフォーマットチェックを行う。

投稿受付工程（Reception）：

投稿ファイルを受け取ると自動的に原稿解析・変換ツールを起動して共通フォーマットのXML文書を生成する。

査読依頼工程（Review Request）：

査読者を選定すると，原稿の文書ファイルフォーマットから作成可能なものの内で，査読者が最も好むフォーマットの出力原稿を作成し出力する。査読者の受取可能な原稿フォーマットは編集管理システムの査読者データベースに格納されており，文書処理時にパラメータとして引き渡される。

英文校閲工程（English Revision）：

英文校閲では予め定められた原稿フォーマットで出力原稿を作成・出力する。

文章編集工程（Document Editing）：

学協会内で本文の編集を行う場合，編集用スタイルに従い，文書ファイルに論文番号や受付日などの管理情報を埋め込んで編集用の原稿を作成してから，ワードプロセッサなどの編集ツールを起動・監視する。また，著者校正ゲラの出力も行う。

レイアウト工程（Layout）：

文章編集工程同様，レイアウト用スタイルに従い，文書ファイルに巻号ページや柱などの出版情報を埋め込んでレイアウト用の原稿を作成してから，DTPなどのレイアウトツールを起動・監視する。著者校正ゲラの出力も行う。

チェックアウト工程（Check-out）：

編集が終了すると，印刷版下を作成して印刷工程に引き渡すとともに，オンライン出版用原稿を作成してオンライン出版システムに転送する。

(2) 文書処理ツールの種類

個別の文書ファイル形式に対応して，工程ごとにこれらの文書処理を実際に行うのが文書処理ツールである。市販のワープロやDTPなどを導入したり，学協会や分野のニーズに応じたツールを開発して組み込んだりすることができる。現在は，MS-Word, FrameMaker, LaTeX, AdeptEditorなどに対応しているが，コマンドラインから起動・監視できるソフトウェアであれば容易に組み込むことが可能である。文書処理ツールの例として以下のようなものがある。

- 著者用，編集用のテンプレートやスタイルファイル
- 投稿用パッケージング・アンパッケージングツール
- フォーマットチェッカー
- 構造化テキスト抽出ツール（投稿票作成用，引用リンク作成用，全文検索用）
- 原稿編集・レイアウトツール
- 原稿出力ツール（査読，英文校閲，著者校正，印刷版下，オンライン出力）

(3) 動作環境

学協会にはすでに各種の計算機が導入されており，また，情報化を実行しサポートする事務局員，編集委員などの経験や知識もさまざまであるため，開発システムはできるだけ多くのプラットフォーム上で稼働することが必要である。

また，サブシステムやツールにはそれぞれ最適な動作環境を選択できるよう，異なるプラットフォームの混在も可能でなければならない。

現時点では，サーバ，クライアントとも，Windows NT/98/95系とSolaris系の2系統に対応しており，LinuxやMacintoshについても今後検討してゆく。

(4) 文書ファイル形式

文書処理ツール群が扱うことができる文書ファイル形式の例としては以下のようなものがある。なお，編集管理サブシステムや文書処理マネージャなどには文書ファイル形式に関する制約はないので，文書処理ツールを追加するだけで異なる文書ファイルを扱うことが可能となる。

- 投稿原稿：MS-Word，LaTeX，XML，紙
- 査読原稿：PDF，紙

- オンライン出版原稿：PDF，XML，HTML（XMLからの自動変換）
- 印刷原稿：PostScript

3.3 システムインタフェース

電子編集システムの機能はすべて，サーバ上で動いている httpd の CGI を通して編集管理サブシステムの制御の下で動作する。しかし，大部分の文書処理機能は編集管理サブシステムからクライアント上の文書処理サブシステムを呼び出して実行する必要がある。

クライアントにはPCやWSなど，様々なハードウェアやOSが並行して用いられる可能性があり，また，情報処理技術の進歩にあわせて新たなクライアントを導入する必要も出てくると考えられる。そこで，サーバ・クライアント間の連携はすべてHTTPとCGIを用いて実現することとした。サーバ・クライアント間の処理の流れの概念図を図4に示す。

管理的な機能は編集管理サブシステム単体で実現できるので，ユーザはWWWブラウザを通して対話処理を行うことができる。クライアントから文書処理ツールを動かす場合，利用者は編集管理サブシステムにアクセスして原稿を選択し，操作を選択する。ここからがローカルユーザとリモートユーザでは動作が異なる。リモートユーザでは，編集管理サブシステムからサーバ上の文書処理マネージャが直接呼び出され，文書処理ツールが起動されて実行結果が編集管理サブシステムに返される。一方，ローカルユーザの場合は，文書処理指示ファイルがCGI経由で送られてくると，ヘルパーアプリケーションとして文書処理マネージャが起動され，そこから文書処理ツールが起動される。文書処理ツールの実行が終了すると，文書処理マネージャがサーバの CGI に結果報告を送ることで処理結果を編集管理サブシステムに通知する。

リモートユーザがクライアント上で文書処理ツールを起動することも原理的には可能であるが，そのたびに文書ファイルをネットワーク経由で転送する必要が発生するため，実用的ではない。そのような場合は文書ファイルをエクスポートしてローカルに一連の処理を行い，結果をインポートする形を検討している。

3.4 文書処理マネージャの機能要件

一つの学協会においても編集工程の処理段階に応じて多様な文書処理ツールを必要としたり，複数の文書ファイルフォーマットが併存するためにツールを使い分ける必要があったりして，同一のハードウェアやOS環境では文書処理要求を満たせない場合もある。また，処理の内容に応じては

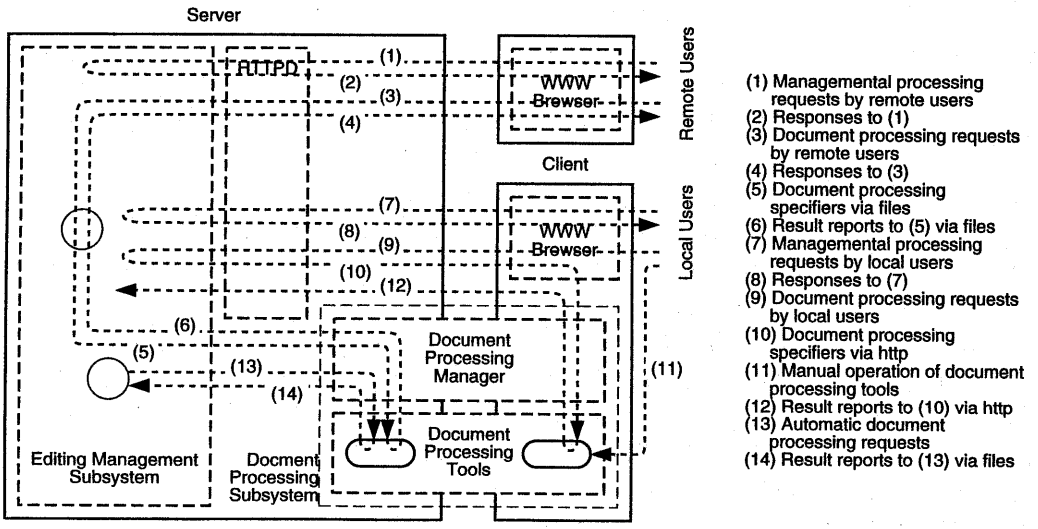

図4 文書処理制御フロー

クライアント上で処理するのが適当な場合と，サーバ上で処理をするのが適当な場合がある。

そこで，文書処理マネージャが原稿実体の文書ファイル形式や処理ツールに依存する部分を局所化してカプセル化し，文書IDと文書処理要求の形で抽象化したインタフェースを編集管理サブシステムに対して提供する。内部では個々の原稿ファイルの格納場所や名前，文書ファイル形式，文書クラス，およびバージョンを管理し，編集作業工程に応じて適切な文書処理ツールを起動・監視する。

この機能の実現のために，文書処理管理方式を新たに開発した。この方式では，文書処理工程，文書ファイル形式，出力文書形式などとともに，改訂原稿の投稿やレイアウト作業などの人的操作によるバージョンの更新を加えた，複雑な原稿実体間の依存関係と，原稿実体間の変換処理手順とをルールとして文書処理定義に記述する。これを解析し，実際の論文原稿に適用することにより，柔軟で高度に抽象化された文書処理機能を提供している。

4. 文書処理定義

文書処理マネージャの動作とインタフェースを定義するのが文書処理定義であり，文書要求定義（Document Requirement Definition）と文書クラス処理定義（Document Class Processing Definition）からなる。

文書要求定義は編集管理サブシステムとのインタフェースを定義するもので，各編集工程において要求する文書クラスとバージョンを指定する。

文書クラス処理定義では，文書クラス間のバージョンも考慮した依存関係と，文書クラス間の変換処理手順，および処理結果の文書クラスインスタンスに対する制御方法を記述する。

文書処理定義の簡単な例の一部をリスト1に示す。この例は，査読用原稿作成と文書編集の定義部分を抜き出して簡略化したものである。最初に編集管理サブシステムから受け取る文書要求識別子と，それに対応した文書クラス名のリストを文書要求定義として記述している（<1>,<2>）。このリストは要求文書クラス名（'@'から始まる）の

リスト1 文書処理定義の例

```
### Document Requirement Definition

# Preparation of Revewing Document
#  Referee's Preference List: %preflist%
RFR_MSC:=@RFR_MSC.%preflist%                    <1>

# Text Editing Operation
#  Word and LaTeX are allowed document file
formats.
EDIT:=@EDIT.WORD(0,+1) | @EDIT.LATEX(0,+1) <2>

### Document Class Processing Definition

# Registration of Contributed Article
@SBM_REG.WORD: ; vers(+1,1) save=yes        <3>
{  ...  }
@SBM_REG.LATEX: ; vers(+1,1) save=yes       <4>
{  ...  }

# Preparation of Refereeing Document
# Omitting a method to generate PDF from LaTeX
@RFR_MSC.PDF::                          # Word => PDF
        @SBM_REG.WORD; vers(0,0) save=yes <5>
{
  word_referee -print distiller sbm-%minor%.doc
  mv sbm-%minor%.pdf rvw-%minor%.pdf
  cp rvw-%minor%.pdf %common_dir%/%ouptut_file%.pdf
}
@RFR_MSC.PAPER::                  # Word => Paper
       @SBM_REG.WORD; vers(0,0) save=yes  <6>
{
  print_form %title% %doc_id% %due_date%
  word_rvw -copy 2 sbm-%minor%.doc
  alert "Two copies are output to printer-0."
}
@RFR_MSC.PAPER::                 # LaTeX => Paper
      @SBM_REG.LATEX; vers(0,0) save=yes  <7>
{  ...  }

# Preparation of Editing File
#  Doc. ID and other info. are
#  incorporated at the first time.
@EDIT.WORD::
        @SBM_REG.WORD; vers(0,0) save=yes <8>
{
  word_imp -style edit "%doc_id%,%accept_date%"\
    submit-%minor%.doc edit-%minor%.doc
}
@EDIT.LATEX::
       @SBM_REG.LATEX; vers(0,0) save=yes <9>
{
  latex_imp -style edit %doc_id%  %accept_date%\
    submit-%minor%.tex edit-%minor%.tex
}

# Revision of Editing File
@EDIT.WORD::
    @EDIT.WORD(0,-1); vers(0,+1) save=yes <10>
{
  cp edit-%minor(-1)%.doc edit-%minor%.doc
  word edit-%minor%.doc
}
@EDIT.LATEX::
   @EDIT.LATEX(0,-1); vers(0,+1) save=yes <11>
{    ...    }
```

ORリストであり，左から順に要求文書クラスの取得が成功するまで評価される。新しいバージョンの文書を要求する場合は要求文書クラス名の後ろに差分を指示する。

査読原稿の作成では，処理対象文書のファイル形式と査読者の受取可能な原稿フォーマットの組み合わせで作成すべき原稿形式を最終的に決める必要があるが，文書要求定義ではまず受取可能な原稿フォーマットへの展開を行っている（<1>）。

また，原稿編集では，現在の文書インスタンスのバージョンよりも一つ新しいバージョンで，編集形式の文書クラスの原稿を要求している（<2>）。

次に各文書クラス間の依存関係と変換処理手順を，文書クラス処理定義に記述する。依存関係は文書クラス名と文書バージョンの組み合わせで構成される。文書クラス名の後に"::"があるもの（例えば<6>,<7>）は，選択的な依存関係，つまり，依存関係が解消できるものが見つかるまで同じ文書クラス名の文書クラス処理定義を試してみることができる。依存関係の後にセミコロンで区切って書かれているのは文書処理制御である。ここには，生成した文書クラスインスタンスのバージョン更新や保存，古いバージョンの削除などの制御を書く。次の行から波括弧で囲まれた部分が文書クラス作成のための変換処理手順である。この部分は各OSが備えているコマンドスクリプトに，文書処理マネージャの組み込みパラメータ（例えば<5>の%minor%）や編集管理サブシステムから渡される外部パラメータ（例えば<5>の%output_file%）の参照機能をつけ加えたものである。外部パラメータ名は編集管理サブシステムから与えられる文書処理要求中のデータタグに対応している。

査読原稿作成の文書クラス処理定義では動作の説明のため，LaTeXからPDFの生成方法を削除してある。査読者の希望する原稿形式がPDF，紙の順であって，文書インスタンスのファイル形式がLaTeXだとすると，まず@RFR_MSC.PDFが評価され，依存関係が解消できないために要求文書の取得に失敗する。次に@RFR_MSC.PAPERが評価され，２番目のルール（<7>）で依存関係が解消でき，要求文書の取得に成功する。

また，文書編集においては，文書ファイル形式がWordの場合，現在のバージョンから新しいバージョンのファイルを取得するためにルール<10>を評価する。しかし，初回は編集用ファイルができていないので，先ににルール<8>が適用されて，投稿ファイルから編集用ファイルが作成される。

5. 動作例

本システムを編集の各工程に使用することの効果を動作例によって示す。

(1) 論文投稿

著者はスタイルファイルやテンプレートを用いて論文を執筆し，パッケージングツールを用いて投稿ファイルを作成する。郵送，電子メール，FTP，WWWなどの手段によって投稿する。例えばWWW投稿では，著者は図５に示すように学協会のWWWサーバにアクセスし，投稿用フォームに投稿ファイル名と簡単な情報を入力し，ファイルをアップロードする。すると，サーバ側で文書処理ツールが起動され，投稿ファイルを展開し，原稿ファイルを共通フォーマットのXMLファイルに変換して編集管理システムに引き渡す。編集管理システムはここから文書管理情報を抽出して投稿票を作成し，著者に確認を求める。著者が必要な修正を行って投稿実行すると原稿ファイルが保存され，管理情報が登録されて仮受付が完了す

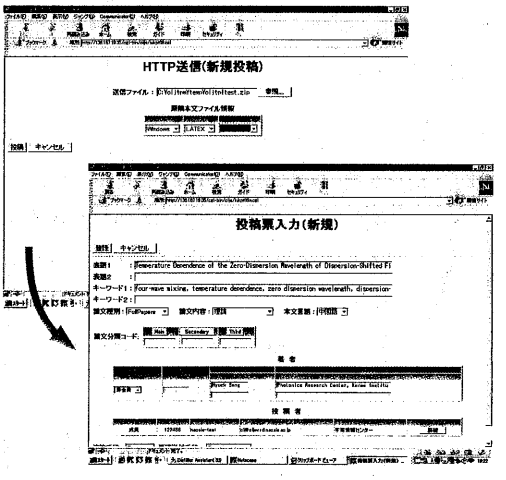

図５ 論文投稿処理

る。

(2) 査読依頼

編集委員はWWWブラウザからインターネットを通じて編集管理サブシステムにアクセスする。査読者を選択して依頼実行すると査読者の受取可能な原稿形式と文書ファイル形式の組み合わせで最適なフォーマットの査読用原稿（紙，PDFなど）が作成され，査読者がアクセスできるようになる。この場合，原稿の作成がサーバ側でバッチ的に実行できる場合は原稿作成は自動的に行えるが，文書処理ツールがクライアント側でしか動作しない場合は事務局員が介在することになる。

(3) レイアウト

編集やレイアウト業務では，操作を指定すると，受付日や柱，原稿番号などの必要な情報が埋め込まれ，実際の原稿の処理に必要なワープロやDTPが起動されるて文書ファイルが開かれる（図6）。操作を終了すると自動的に管理情報が更新される。

6. おわりに

本稿で紹介したプロジェクトは現在も開発が進行中であり，システムの実用化までには状況の変化に対応した変更もあり得る。現在，初期の開発がほぼ完了した状況であり，今後は協力学協会に試験利用していただき，実用に耐えるものに改善してゆく予定である。その後，一般の学協会や大学等の学術機関へソースコードごと提供し，学術目的の範囲で自由に改変して利用していただきたいと考えている。

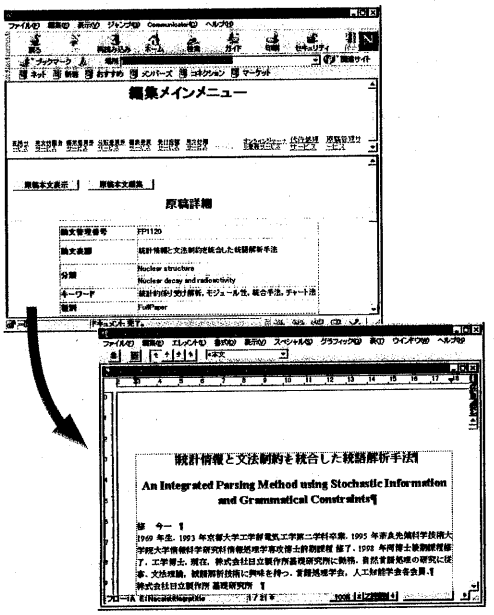

図6 レイアウト操作

一方，多国語や外字，数式や化学式などの表現形式に関する拡張や，情報技術環境の変化，印刷物中心からオンライン中心への出版形態の変化への対応などに関する開発は継続して行ってゆく予定である。また，オンラインジャーナルならではの付加価値として，引用文献へのリンクや被引用の逆リンク，マルチメディア情報の提供，あるいは著者や読者からの追加情報の提供などに必要な機能拡張も並行して進めてゆく予定である。

参考文献

[1] 大山敬三, 神門典子, 佐藤真一: NACSISオンラインジャーナルプロジェクト, 情報の科学と技術, vol.49, no.6, pp.295-300(1999).

[2] NACSISオンラインジャーナルプロジェクト システム開発のページ, http://www.rd.nacsis.ac.jp/olj/index-j.html

[3] NACSISオンラインジャーナルプロジェクト XML DTDの入手申込, http://www.rd.nacsis.ac.jp/olj/access_dtd-j.html