2003－CE－70　(6)
2003／7／12

人間をモデルとして拡張したオブジェクト指向ファイル概念とその思想
x-file、x-folder、virtual-router とは

水島賢太郎
神戸女子短期大学・初等教育学科
mizusima@kwjc.kobe-wu.ac.jp

要旨:

情報社会では、文書ファイルのコンピュータ処理やネットワークを通しの交換が増える。このことを誤り無く行なうには、そもそも元の文書ファイルが正しく構造化されている必要がある。そこで正しい文書ファイル作成を補助するための「電子原稿用紙」と名づけたの仕様を検討してきたが、その過程で開発するソフトが扱う文書ファイルとして従来のファイルでは不十分だという結論に至った。そこで、人間のような振る舞いをするオブジェクトをモデルとして新しい概念のfile仕様を考え、x-fileと名づけた。同時に、x-fileを有効に利用するための仕組みについても考察した。

The idea of extended file based on the concept of humanlike object
x-file、x-folder、virtual-router

Mizusima Kentaro
Kobe Women's Junior College

Abstract:

In the information society, many document files are processed by computers and exchanging through a network. In order to carry out such process correctly, the document files must have the right structure. I have continued the consideration about the specification of the new software named "electronic manuscript paper" which supports documents creation with the right structure and found out that the existing document files specification is insufficient for it. So, making reference the object, which carries out behavior like man, I produced the idea of new file specification. The new file is named “x-file”. Simultaneously, the structure for using x-file effectively was also considered.

1. はじめに

1.1 研究テーマ

本研究が目指すテーマは、事務現場での文書作成の効率化と教育現場での電子化文書の教育を正しくかつ分かりやすく行なうための「電子原稿用紙」ソフト開発に向けて、そのソフトの文書ファイル仕様について検討である[1]。

ここでいう効率化とは、①軽快な操作インターフェイス、②ファイル管理の容易さ、③異なったコンピュータ環境でも同一ソフトがあれば常に同一環境で作業できるというポータビリティ、を意味する。また教育現場での分かりやすさとは、文書の論理構造と物理表現とが明瞭に分離された形で可視化されているということを意味する。

さて、普通文書ファイルといわれるものを分類すると、①ワープロ文書ファイルのように特定のワープロに対応した複雑な仕様のバイナリーファイル、②これと対極をなす文書構造情報を持たない単なるテキスト文字のみのテキストファイル、③それらの中間のようなHTMLやXMLファイルのような「文字コード＋ブロック構造ルール」型のテキストファイル、が考えられる。本稿で提案する文書ファイルは③のタイプにあたる。したがって皮肉なことだが、バイナリーであるワープロの文書ファイルは、本テーマでいうところの文書ファイルではない。

以下、本稿の記述を簡潔にするため、幾つかの言葉を仮称として導入する。

- **x-file、x-folder、virtual-router**

  電子化文書ファイルを「**拡張された文書ファイル**

Mizusima　Kentaro
Kobe Women's Junior College

(extended-file)」、それを管理･運用するための仕組みを「**拡張されたフォルダ** (extended-folder)」、「**仮想的なルータ** (virtual-router)」と仮称し、それぞれを、**x-file、x-folder、v-router**と略記する。

1.2 テーマアイディアの位置付け

x-fileのアイディアが最終的に目指すのは、あたかも人間や自立型ロボットのようにコンピュータシステム内で同一テーマを扱っている他のx-fileと相互に連携を取りながら文書作業の効率をあげるというものである。また、x-fileの仕様には、たとえば図形ソフトで作られた画像ファイルをその内部に格納できるようになっているので一種のコンテナのようなものともいえる。

x-fileのアイディアは、ある意味でXMLが目指す世界の「一部」と見ることもできるし、また、XMLを部分として「含む、あるいははみ出している」ともいえる。

「一部」という意味は、XMLが目指す壮大な電子化文書処理の世界の一部、すなわち文書作成と閲覧に限定された「電子原稿用紙」ソフトでの利用を目的としているからである。一方、「含む、あるいははみ出している」という意味は、x-fileのアイディアがHTMLやXMLといったマークアップ言語ファイルや電子メールファイルの仕様から有効と思われる概念を抽出してまとめあげているということと、またファイルがたとえば自己解凍型圧縮ファイルのようにある種のメソッドを持って自律的な能動性(「文書データ＋メソッド」)というオブジェクト指向の概念を取り入れているからである。

「独立」という意味は図1のようにXML文書ファイルがその活用･利用という世界を目指した仕様であるのに対し、x-fileでは電子原稿用紙ソフトがいかに効率的にそのファイルを作成できるか目指した仕様だからである。

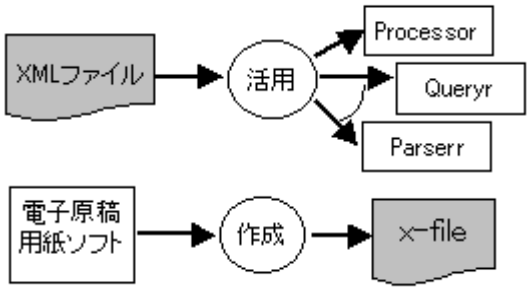

図1 XMLファイルとx-fileの目指す方向

注意すべきことは、インスタンスとしてのx-fileは電子原稿用紙ソフトで作成されるものではあるが、x-fileの仕様は、HTMLの仕様がブラウザの設計を規定したように電子原稿用紙ソフトの設計を規定することである。

2. テーマの設定の背景

十数年前のオフィスと今のオフィスを比べると、パソコンの普及とネットワーク環境の整備がもたらした影響の大きさを痛切に感じられる。では、影響はすべて事務作業の効率化に繋がっているのであろうか。ここでは本研究のテーマである電子化文書に絞って簡単に考察しておく。この考察の目的は、x-fileの仕様やx-folder、virtual-routerといった概念を考えるに至った理由を示すためである。

2.1 オフィスにとっての文書作成作業とは

2.1.1 オフィスのワープロ作業の実体

高機能ワープロとプリンターの改良により、ほんの十数年前には見られなかった高品質のワープロ出力書類が、割合においても絶対量においても増加した。

では、このような環境は事務文書作りの効率を上げているのだろうか。なるほどワードに象徴される高機能ワープロには「美しい見栄えの文書」を「手際よく作成する機能」が多く実装されている。しかし、何パーセントの人間が「手際よく作成する機能」を「**効率的**」に使いこなしながら「**正しい**」文書ファイルを作れているのだろうか。実態は、しばしば思いがけない動きをするワープロを騙し騙し使い、圧倒的に多数を占める社内文書にたいしても不要なまでの飾りを行い、かつそのために無駄な時間を費やしているのではないか

ここで「**効率的**」とは、多くの文書作成ソフトに含まれる構造化文書作成機能と物理表現上の機能の差異を明確に理解し、それを適切に駆使して文書作成できる能力を意味する。また、「**正しく**」とは出力された文書が紙メディアレベルで構造化された物理表現(見栄え)を持っているか否かではなく、その電子化文書がコンピュータによって適切に処理･加工でき、かつ、ネットワークで効率よく適切に共有や再利用できるファイルであるか否かを意味する。

2.1.2 ワープロの進化1 (タイプライターの子孫)

ワープロ出現以前、表音文字文化圏オフィスの文書作成の効率化はタイプライターによって支えられていた。タイプライターは手動から電動へと進んだ、その本

質はハードな機械であった。このため、修正（タイプミスと文書構成の変更）の困難が文書作成効率化のボトルネックとなった。

文書作成の効率化にとってワープロの最も重要な寄与は、この文書作成時における修正問題の解決といえる。実際、事務現場で作られる文書に求められる重要なポイントは内容の明晰さであり、商業出版文書レベルの見栄えは一部客向けのパンフレットを除けば、1種類のフォントしかないタイプライターで出力されるレベルで十分といえる。

もし事務文書作成が多くの事務現場でみられるような紙出力だけを目的とするならば、エディタより少し見栄えのよい程度の印刷機能を持ったワープロ、たとえば昔あった「松」や一太郎Ver.3 レベルのワープロの方が、オフィスの文書生産性にとっては望ましいといえる。

さらに高機能ワープロの勉強内容の誤りも多い。すなわち、スタイル設定やアウトライン機能、テンプレートといった項目はほとんど学習対象から外され、相変わらず紙の見栄えのために分厚い参考書で時間を無駄に費やしているのが、事務現場の実態といえよう[註1]。

高機能ワープロがもたらした最大の問題点はここにある。また、論理構造作成機能と物理表現機能をはっきり峻別していな操作メニュー設計は、初学者に「うまく使えないのは自分に原因がある」と思いこませ、設計改善への声を摘み取ることになる[註2]。

2.1.3 ワープロの進化2 (タイプライターとの決別)

既報のようにマイクロソフト社はワードの次期バージョンとしてXML対応を高める。このことは、ワープロが明確に紙メディア出力中心の高機能タイプライターから、進化の質を変えたことを意味する。

この進化はWebページ作成支援ソフトとのある種の融合を意味する。実際、ワードには（品質はともかく）Webページに変換するファイル保存がサポートされている。そして、Webページ作成支援ソフトが紙文書のページ概念を取り入れれば、両者の境界はより曖昧になる。

2.2 文書ファイルの管理　メタ文書問題

ソフトを使った文書実務では、文書ファイル作成同様、ファイル管理が重要になる。管理問題にはたとえば次のようなものがある。

2.2.1 バージョン管理とバックアップ (ファイルヘッダー問題)

簡単な事務連絡文書以外の報告書や論文では、何度もの校正が必要で、かつ完成までに日数を要する。この作業で問題になる事柄を見てみよう。

- **バージョン管理**

  私自身の文書作成経験を振り返ると、文書作業では必ずしも一つのファイルを上書きするのではなく、最終稿までの色々な段階でのファイルを別名保存し、時には古いバージョンに立ち戻って参照したりすることがある。また、自宅と職場を行き来しながら仕事をしていると、最新のファイルで作業しているつもりが古いファイルをいじくっていたという悲劇も起こる。

- **バックアップ**

  これとは別に、物理的にもソフト的にも何時突発的にクラッシュするか分からない状況では複数場所へバックアップが必要になるが、このときでも前述同様の問題が起こる。

このように、ファイルの同一性（アイデンティティ）の保証をいかにとるかは実務現場では重要である。多くの場合、この管理はファイル名と時間情報を参考にしながら人手をかけて管理されるが、市販のワープロに組み込まれた時計は結構狂うというように結構厄介である。この問題が私だけの問題でないことは、バックアップの同期や時間調整のためのソフトがあることからも分かろう。

結局のところこれらの問題は、文書ファイル管理のためファイルヘッダー（メタ文書情報）仕様の貧弱さに起因する。実際、現行の各種ファイルのヘッダーはユーザーが簡単に記述できるものではなく、かつ隠蔽されている。このため、ファイル管理の多くが人間の手を煩わせるのである。

2.2.2 複数のファイルの一元管理

ワードには図といった別のファイルを扱っているファイルデータを取り組む際、その文書ファイルに直接組み入れる「挿入」と、間接的に取り入れる「リンク挿入」がある。リンク挿入はまさにWebページのリンクと同様で、「挿入」はいわば電子メールの添付ファイルと似たもの

である。

さて、「リンク挿入」では親となるファイルとリンクされるファイルとの管理問題が常に起こる。すなわち、リンク切れ問題が起こる。

3. x-filerの概念

x-fileのアイディアは、前章で簡単に考察した電子化文書作成時の諸問題の解決に向けたものであるが、そのアイディアの幾つかは一般的なファイフでにも拡張可能と考えられる。

3.1 x-fileの基本構造

一般的に、ファイルは内容データのブロックとそのファイルのアイデンティティやそれを利用するソフトに関する情報（メタデータ）を書いたヘッダーブロックから成り立っている。このためヘッダー情報は特定のOSやソフトと強い依存性を持つのが普通である。

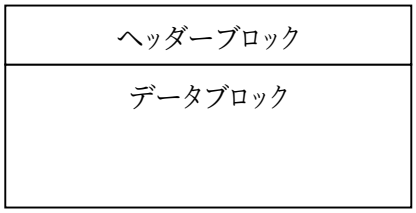

図2　一般的ファイルのブロック構造

x-fileの実体は図3のように、ある約束にブロック構造を（できる限り）テキストデータとして埋め込むことでこの依存性を避けるようにしている。つまりX-fileの実体は、通常のテキストファイルのデータブロックに、それ自体で独立したヘッダーブロックとテキストデータブロックを入れ子として持ったものに過ぎない。

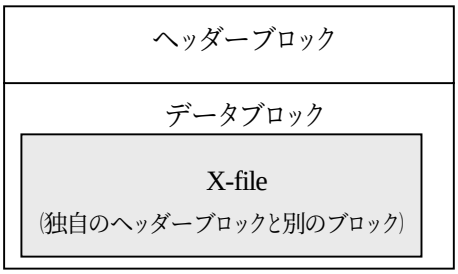

図3　テキストブロックに入れ子されたX-file実体

このアイディア自体は特に目新しいものではなく、HTMLやXMLあるいは電子メールのファイルと同様のマークアップ言語によるブロック構造の仕様と同じ発想に立っている[註3]。

したがって、x-fileの仕様を考えることは、結局はHTMLやXMLのように、前項で述べた諸問題の解決のための仕組みを持ったブロック構造を考えることに他ならない。以下、図3の斜線部の内部ブロック構造を細かく見ていく。

3.2 x-fileのブロック構造（人間をモデルに）

社会生活を営む人間の行動を、役所での住民票発行を例に簡単に見てみる。

窓口に行った人間は、まず自ら自身を証明するID情報（メタ情報ブロック）を提示し、次に処理したい事柄（データブロック）を窓口係員とのやり取り（プロトコル、メソッドなど）によって所期の目的を達する。

x-fileは上記のように異なった内容に応じてブロックを作りながら構造化されたものである。具体化にあたっては、①HTMLやXMLといったマークアップ言語、②オブジェクト指向言語、③電子メールのブロック構成などのアイディアを抽出しながら再構成していった。以下、取り入れたものを簡単に紹介する。

1. マークアップ言語

電子化文書教育では意外なことにあまり強調されないが、HTMLの普及がもたらした画期的な影響に「データファイルがソフト開発を規定する」というものがある。すなわち、たとえばワープロのソフトの開発では、求められる文書表現や編集機能を実現するためにデータファイルの構造が規定されるが、HTMLではこの逆が起こったということである。ワープロの文書ファイルでは上位互換はあっても下位互換あるいは後方互換がないといったソフトのバージョンアップに伴う文書ファイル互換問題をみればこのことは明らかだろう。同様、教育で強調されないものに、ヘッダーブロックにテキストデータとして各種メタデータを埋め込み、Webページファイルの OS依存性を低減させていることがある。

x-fileでは「各種ブロック構造をテキストで埋め込む」ということこのようなアイディアを採用し、ユーザーに対するファイル構造の可視化を高めて、教育的にも取り扱いやすい仕様を目指している。

2. オブジェクト指向言語

電子原稿用紙で扱う文書は、当然ながら紙出力だけが目的ではなくコンピュータで処理される電子化文書である。その際、Webページに象徴されるように一つのページには各種のオブジェクトがリンクされる。ここで問題となるのがリンク切れやバージョン変更問題である。

リンク切れには保存場所の変更やファイル名の変更、バージョン変更などがあり、バージョン問題はファイル内容変更に伴う同期問題がある。

x-fileの仕様では、オブジェクト指向プログラミングのメソッドにあたるものを埋め込むことにより、ファイルが自立的にこの問題の解決できることを目指している。その際、HTMLファイルのヘッダーにJavaScriptといったスクリプトの埋め込み方法が参考になる。

3. 電子メール

電子原稿用紙の文章作成モードでの図表の取り扱いは、単にタイトルしか書かない（けない）[註4]。そして、完成した文書の閲覧や印刷では、図表自体とタイトルとをリンク関係で取り扱うのである。その際、図表を置く場所として、①x-fileに埋め込む、②x-fileとは別ファイルとしてフォルダに置くというやり方を考えている。

①で参考になるのが、電子メールでの添付ファイルの取り扱いである。電子メールの添付ファイルは、本文と添付ファイルは別領域（ブロック）で管理されているが、この考えをマークアップ言語の考え方で管理するようにするわけである。②はそのブロックにURIのようなものを置くことにより実現させる。

3.3 x-fileの機能別ブロック構造

前節で見た事柄を、x-fileは右の図4に示す5つの管理ブロック構造によって実現を図っている。以下、各ブロックの詳細について紹介していく。

なお、以下の議論で見られる各種機能には、すぐ実現可能なものから実現の困難なものまで、いわば玉石混淆状態と考えられる。したがって各ブロックの機能の実現化にあたっては領域的取り扱いも考えられる。

3.3.1 ヘッダーブロック

このブロックは文書内容に対するメタデータを保存部するブロックで、2.2.1 で述べたファイル管理問題解決のために必要なデータが書かれている。書かれるデータは、ユーザー自身で追加･訂正できるものとソフトが自動的に追加･訂正するものに分かれる。ユーザーは後者データの閲覧はできるが直接更新は出来ないものとする。以下,ヘッダーブロックに帰されるものを幾つか示してみる。

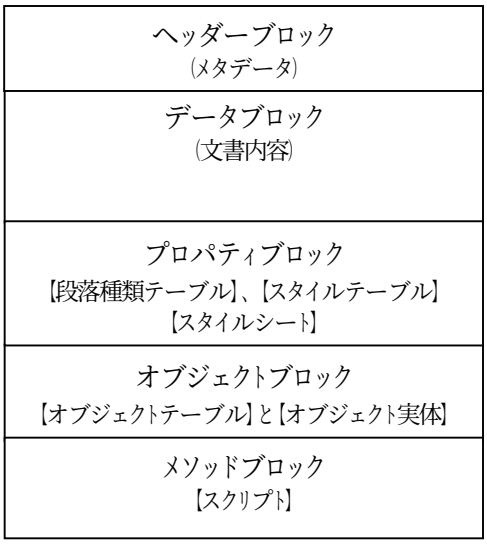

図4 x-fileのブロック構造

1. x-file 保存場所（フォルダ）情報

文字通り、そのx-fileを保存した場所の情報をOSのフォルダ管理情報から入手して書き込まれるもので、後述v-routerを通して変更あるごとに自動的に書き換えられる。この情報は別のx-fileと共にv-routerのデータテーブルにも送られる。

2. x-file 名

x-fileはx-file名（名前＋バージョン番号＋拡張子）によって文書ファイルのアイデンティティを管理している。したがって、x-fileを入れ子とする親のファイルヘッダーのタイムスタンプが異なっていても、x-file名が同じなら同じファイルと認識する。つまり、x-fileではタイムスタンプはファイル同一性にとって参考程度の情報に過ぎない。バージョン番号はファイル内容が変更されるごとに自動的に更新される。たとえば電子原稿用紙ソフトでの操作イメージは次のようになる。

⑴ **上書き保存**
文書内容に変更があると自動的にバージョン番号が更新され、新しいものだけが保存される。

⑵ **名前を変えて保存**
バージョン番号1の新しいものが保存される。

⑶ **バージョン保存**

バージョン保存では、編集していた元のファイルを残したまま、新しいバージョン番号を振られたファイルが保存される。最近の電子メールソフトに見られる「同一件名のやり取り時に自動的に連番が振られる」といった機能に似ている。

3. バックアップファイルテーブル

複数箇所にバックアップしながら仕事をしている場合、バックアップ問題が起こる。そこでx-fileではヘッダーブロックにはバックアップファイルのアドレスに対応する情報を「バックアップファイルテーブル」として置くようになっている。

このテーブルとして次のような仕様が考えられる。

各x-fileは、他のx-fileと後述のv-routerを通してテーブル情報の交信し、テーブルの同一性を確認する。このとき、もし自分より古いバージョンや新しいバージョンデータを持っているx-fileが見つかると、自分を保存しているx-folder（これも後述）にそのx-fileをコピーし、同時にテーブルを書き換える。この機構を通して、一つの文書業務中は、その業務が完了するまですべてのx-fileとx-folderの状態は同一状態に保たれる。この意味で、同一グループにとってすべてのx-fileは同一のものとなる。

● **想定される問題**

私たちはバックアップ用にフロッピーのようなリムーバブルメディアを使う場合がある。この場合、①もし装着を忘れたらどうなるか、②メディアの容量によってはすべてのファイルをコピーしきれない場合はどうするか、という問題が起こる。

①の場合は、それを利用する電子原稿用紙ソフトにメッセージが出て、v-routerとの交信により復元されればよい。問題は②の場合だが、その場合はテーブルのみの同一性だけを保ち、当面不要とユーザーが考えたファイル群のテーブル領域に未利用チェックを入れる。

4. ユーザー設定情報あるいはテンプレート

x-fileを出先にある電子原稿用紙で取り扱う際、通常の環境と同じようにできることが望ましい。これを実現するには、ウインドウズでいうiniファイルや文書テンプレートが必要となる。これらの情報はたいして大きくないのでx-fileのヘッダー部に置くことが望ましい。実際、私自身、ワードの文書を出先で扱う場合に自分の環境と異なっていて困ったこと経験を多くしており、また、下手に他人のワードをカスタマイズすると、その内容がnormal.dotに残ってしまって人様に迷惑をかける恐れもあるからである。

当然ながら、このヘッダー情報は自分が普段使うマシンには共通のデータ保存フォルダか、もしくは電子原稿用紙ソフトのしかるべき場所に保存されている。

この情報は、前項のx-file名での議論同様、同一グループにあるすべてのx-folderと同期を取って同一性の保障をとる必要がある。

3.3.2 データブロックとプロパティブロック

x-fileは電子原稿用紙ソフトの文書ファイルである以上、最低限度、文書構造を保持できる仕組みが必要となる。また、電子原稿用紙ソフトにはアドオンもしくは最初の仕様として簡単な物理表現機能を持たせたい。

また、文書を作成する方法はワープロのような「シンプルな」操作でありながら、出来上がったものはHTMLやXML整形文書のようなものというイメージである。ここで「シンプルな」とは、ワードで文書を正しく作る手順、すなわち、「先ず文字を入力して段落を作り、次にその段落の種類をチェックする。見栄えはスタイル（S）ウインドウで別途作っておき、それを段落スタイルとして適応する」といったイメージである。この「シンプルな」操作で出来上がった原稿は、HTMLやXMLの整形式文書ファイルのようにタグがべたべた貼り付けられていな文字列でありながら、タグによって果たされる文書の構造化や物理表現を実現しなければならない。

1. 求められる機能の分解

この目的を果たすには、HTMLやXML整形式文書からタグを抜き出し、文書内容のテキストと別々に管理できる仕組みが必要となる。

ここではある文書モデルを考え、次にそのモデルから文書内容とタグを分離し、幾つかのブロックとして独立に扱えないかを検討した。

● **文書モデル1　単一段落**

まず文書を異なった機能（構造化要素）を持った段落ブロックのように積み重ねたものと見る。これを文書積木ブロックモデルと呼ぶ。

文書積木ブロックモデルは、段落ブロックをP_iと表す

と次のような式で表される。

(1)　文書＝$P_1+P_2+\cdots+P_i+\cdots+P_n=\Sigma P_i$

さて、文書を原稿ではなく構造化要素と物理的表現（スタイル、STと略記）を含んだ最終完成品として見た場合、各P_iは次式のように表される。

(2)　P_i＝構造化要素$(k)_i$＋文章内容＋$ST(k)_i$

ここで構造化要素(k)のkは本文、引用文、見出し文といった種類を表す。

段落を式(2)のように見ると、次の関係ができる。

(3)　文章内容$_i$⇔構造化要素$(k)_i$⇔$ST(k)_i$

(3)の関係は、あたかも「i」をポインターのように使うことにより、段落の文章内容であるテキストデータと、その段落ごとの構造化要素の種類、見栄えのスタイルを別々のブロックとして管理できることを意味している。

すなわち図4のデータブロックには「文章内容$_I$」を単純なテキストとして格納し、プロパティブロックには構造化要素$(k)_i$のテーブル（＝【段落種類テーブル】）とその段落に適用すべきスタイル$(k)_i$のテーブル（＝【スタイルテーブル】）を独立したブロックとして配置できるわけである。

なお【スタイルテーブル】はスタイルの種類(k)だけからなるテーブルで、その内容はx-fileの別ブロックもしくはX-fileとは独立した外部ファイルにスタイルシートとして書くものとする。

● **文書モデル2　複数個からなる段落**

実際の文書には、章や節あるいはリストというように、複数の段落を持つものが存在する。式で表せば次のになる。

(1)　$B_m=\Sigma Pi$

例えば、第3節が、文書のはじめからみて 100 番目の段落から 150 番目の段落なら次のように対応する。

(2)　$B_3=P_{100}+P_{101}+\cdots+P_{150}$

すると、次のような対応関係に分解できる。

(3)　$B_3(100,150)$⇔機能(3)⇔スタイル(3)

プロパティブロックの【段落種類テーブル】と【スタイルテーブル】には、このような対応関係に基づいたブロックも入っている[註5]。

● **段落内要素の取り扱い**

HTMLにはem要素といった特定の文字列を修飾する要素があり、またワードには文字スタイルというものがある。これの取り扱いは「文書モデル2　複数個からなる段落レベル」と同様の方法が考えられる。すなわち、文書の先頭からの文字数番号として処理すればよい。

当然ながら、対応関係に応じた処理や書き込みは電子原稿用紙ソフトの仕事である。

3.3.3　オブジェクトブロック

オブジェクトブロックは、図や表といったオブジェクトを取り扱うブロックである。

電子原稿用紙ソフトの文章入力モードは、エディタのように単なる文字入力だけを行なう。したがって、図や表はWYSWYGに形では入力せず、単に「図3 x-fileブロック図」のよう入力する。これに対応して、データブロックも単なる文字テキストが入っているだけで、図や表の実体はオブジェクトブロックで管理される。

データブロックにある「図3　x-fileブロック図」といったものとオブジェクト実体は、前節同様の考え方でオブジェクトテーブルとのリンクを介して取り扱うものとする。

なお、オブジェクトの置き方はx-fileに直接埋め込むやり方と外部にあるファイル参照という2つのものをサポートするが、データブロック側ではその違いは意識されず、リンク先のオブジェクトテーブルに書かれたデータの形で振り分けるようになっている。

オブジェクトをx-fileの内部に埋め込んでおくやり方は電子メールの添付ファイルの扱いに似たものとする。また、外部はオブジェクトテーブルにURIのようなものを書いておいて、再度リンクを通して参照に行くという形になる。

3.3.4　メソッドブロック

メソッドブロックはオブジェクト指向プログラミングにおけるメソッドにあたるプログラムを置くブロックである。プログラムには、①いわゆるマクロプログラムが置かれるブロックと②後述のx-folder、virtual-routerとのやり取りのためのスクリプトが置かれる。

ここで①は特に議論はないであろう。

②は 3.3.1 で議論した事柄を実現するためのスクリプトで、x-fileアイコンがフォルダ間移動のためクリックされたりしたとき、あたかも自己解凍プログラム（あるいはウイルス？）のように動きを始め、v-routerを通して他の

x-fileと連絡を取り合うというものである。したがって、このスクリプトx-fileの一部としてユーザーが変更できない隠蔽された固定プログラムである。

メソッドブロックはクリック等の操作だけで実行されるため、ある種のセキュリティ対策を講じておく必要があろう。メソッドブロックに関しては、この問題を含めそれが軽いものとして実装可能か等、いろいとな検討が必要といえる。

4. x-folderの概念

x-folderは、電子メールのMboxの考え方を使って、複数のx-fileや外部オブジェクトからなる比較的大きな文書を取り扱おうというものである。すなわち、電子メールソフトではあたかも個々のメールを一つずつ扱っているように見えるが、その実体は一つのファイルに過ぎず、適当な管理テーブルファイルを用いて、その一部を切り取ったりすることが簡単にできるということにある。

x-folderもx-file同様、後述のv-routerとのやり取りを通して自立的に自己管理できる構造を持たせる。

5. v-routerの概念

インターネットにおけるルータと似た機能を果たすもので、x-fileやx-folderは、それが利用されるごとにv-routerを介して他のx-fileやx-folderとやり取りを行なう。機能には色々考えられるが、たとえば個々のx-fileを保存する場所が別のfolderに移動したり、また削除されたり、フロッピーのように装着されていなかった場合、その変更内容をルーチンテーブルとしての管理がある。これにより、x-fileのヘッダーブロックの仕様が単純になり、また、電子原稿用紙がファイルを呼び呼び出したりする際に「さて、あのバックアップファイルはどこにしまったのか」といった検索の手間が軽減される。

また、v-routerはURLのようなアドレス管理機能をもっており、それのアドレスとx-file名を指定することにより、ネットワークを介してx-fileの処理が行なえるようになっている。

v-routerは、いわばx-fileを参照しようという場合の総合窓口といったものである。

6. 今後の課題

今回報告したアイディアは、「作･文書」教育のあり方の実践と文書管理がへたな私自身の文書作成経験から「こんな電子原稿用紙ソフトがあったらいいのになあ」ということを考えている過程で「ソフトの仕様に先立ってファイル構造の仕様のほうが重要ではないか」と思い至ったことから生み出されたものである。

これからの課題は、今回報告したx-file、x-folder、virtual-routerの概念をもとにして、既報としてごく簡単なアイディアしか示せていない「電子原稿用紙ソフト」の具体的な仕様作成がある。

しかし、私自身の専門は教育であり、一般情報（処理）教育としてのワープロやHTMLの経験はある程度積んだが、情報科学に関して専門家ではない。このため、今回の考察には多くの問題があると思われる。

本報告に興味と関心のある専門家の方々から明らかな誤解や技術的問題の指摘、またアドバイスがいただければ幸いである。

註

註1. 紙出力への見栄え中心のワープロ操作教育は、このような事務員を拡大再生産している。

註2. 私は、応用ソフト学習の困難さの原因が操作インタフーイス設計に起因するか否かを学ぶことを「ソフトのクリティカル･リーディング」呼んでいる。すなわち、ソフトに対する能動的な働きかけを持った学びである。「ソフトのクリティカル･リーディング」はそれ自体情報教育の興味深いテーマであるがここでは触れない。

註3. 電子メールの実体を「マークアップ」というには抵抗があるかもしれないが、< >といったタグでの明示化はないが、発想自体はマークアップ言語と同じといえる。

註4. HTMLでは表をbodyという、いわば原稿用紙部分に置かれるが、表の表現は複雑なため原稿としての可読性が低い。このため、電子原稿用紙では、表を図と同様のものとしているわけである。

註5. 蛇足ながらいえば、HTMLのh要素は、実は章といったブロックを示す「見出し表示」段落に過ぎない。章をブロックで示すには次のイメージのようなタグ構造がいる。

```
<章>
  <h1>第１章　初めに</h1>
        :
</章>
```

同様、ブロックレベルに分類されるhr要素は、どう考えても見栄えの物理要素に過ぎないので、むしろインライン要素と考えたほうがいいのではと思われる。

参考文献

[1] 水島賢太郎、「作･文書」教育のための電子原稿用紙の仕様について、情報処理学会、情報教育シンポジウムSSS2002 論文集、5b-3s、2002.8

[2]水島賢太郎、教科『情報』施行後の大学一般情報教育思想の総括と変革への展望、情報処理学会第 65 回全国大会論文集、Vol.4、P227-228、2003.3