

OS/omicon 第4版におけるデータ管理構造の設計と実現

1 F-7

高野了成, 早川栄一, 並木美太郎, 高橋延匡
(東京農工大学 工学部)

1. はじめに

近年, 計算機の性能の向上とともに, 計算機で扱うことのできる情報も多種多様化してきた。紙とペンのインタフェースによる手書き情報などのパターン情報もその一例である。パターン情報のように多様な形態をもつ情報を計算機上で扱うには, その情報とそれに対する操作をオブジェクトとして抽象化して管理することが有効である。筆者らの研究室では手書きインタフェースなどでの研究基盤である OS/omicon 第4版 (以下, V4) を研究・開発している。本稿では, V4 におけるデータ管理構造の設計と実現について述べる。

2. 従来の FS のリンク機構の問題点

従来の UNIX 型の FS は木構造のファイル空間を持っている。基本的にファイル空間内の名前はファイルの実体と一意であるが, ハードリンク, シンボリックリンクといったリンク機構 [1] を用いることでファイルの実体に対するエイリアスを得ることができ, ファイルの共有が可能になる。

このようなリンク機構を用いた場合, ファイル削除などの操作によって, リンク先の実体が存在しないなどリンク間の整合性が破綻する可能性がある。そこで, ファイル操作に対して自動的にリンク間の整合性を保つ機構が必要である。

3. 「意紙」の概念

V4 は計算機上に紙を仮想的した「電紙」を提供する。この電紙の特徴の中でも, 電紙間のリンクに注目し, 「意紙」を設計, 実装した。

(1) 「意紙」間のリンク構造の管理

紙では文章の書かれた紙の上に図表を書いた紙をはり付けることが可能である。「意紙」はこのように「意紙」間にリンクを張ることで構造を持たせる。また, 「意紙」に対する移動, 消去といった操作に関連する「意紙」すべてに通知し, リンク先の「意紙」が存在しないなどの不整合が起こらないようにリンクを動的に修正する必要がある。

(2) 「意紙」が持つ属性の管理

例えば, 手書き文字は文字認識によって文字コードに変換することができる。このようにオブジェクトは複数の属性のインスタンスである可能性がある。本稿ではオブジェクトの内容を表す属性情報を特に属性と呼ぶことにする。そこで「意紙」は複数の属性を持つことができ, 属性間で相互に属性を変換する属性変換機構を提供する。

4. 「意紙」管理機構の設計方針

(1) 「意紙」間の動的な変更が可能なリンク構造

UNIX のファイルはファイル名と i-node の対応によって名前付けされており, ファイル名は名前空間を階層的に分割した木構造の名前として表される。一方, 「意紙」間の関係は木構造に限定されないネットワーク構造で表される。そして, 各「意紙」からリンクを張った「意紙」を知ることができ, ある「意紙」の操作による影響がリンク先の「意紙」にも関係する場合は, リンクを動的に変更することができる。

(2) 属性変換機構

「意紙」は生成時には一つの実体を持っている。実体が格納されたセグメントはメモリ上にマッピングされており, これをメモリオブジェクトと呼ぶ。ここで, 属性変換を実行した場合, 新しい属性を持った実体が得られ (図1), どちらの実体も一つの「意紙」として扱うことができる。

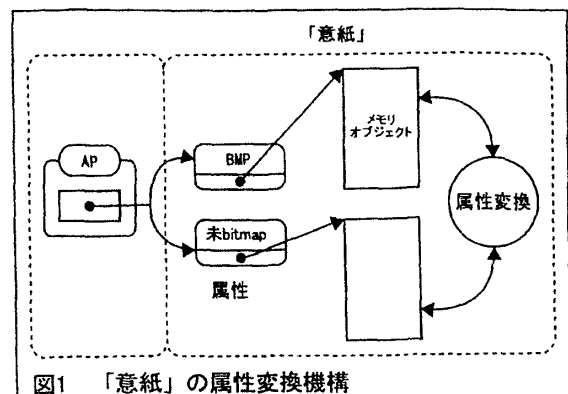


図1 「意紙」の属性変換機構

5. 「意紙」管理機構の設計

5.1 全体設計

「意紙」管理機構はリンク管理部と属性管理部からなる。また、AP から「意紙」へのアクセス法としてポインタというプリミティブなもので提供するが、ユーザが属性ごとに特化した操作を追加できるようにする。

5.2 アプリケーションインタフェース

AP からの使用例を図 2 に示す。「意紙」間のリンクはネットワーク構造をしているが、便宜上ルート「意紙」を設け、ルートからのリンクの経路をたどる名前によって AP から「意紙」を指定する。「意紙」をマップすると、実体へのポインタを得ることができ、ポインタ操作によって「意紙」にアクセスすることができる。マップ、アンマップ、リンクなどの基本的な操作（表 1）はすべての属性が持っている。

```

UBYTE **ポインタ;
INT i;

/** 意紙をメモリ上にマップする **/
ポインタ = 意紙のマップ("テスト");

/** 意紙の実体の中身を表示する **/
for(i = 0; i < 100; i++){
    printf("%c", (*ポインタ)[i]);
}

/** 属性変換する **/
ポインタ = 意紙のリマップ(ポインタ, SHIFT_JIS);

/** 二次記憶に書き込む **/
意紙のフラッシュ(ポインタ, 意紙のサイズを得る(ポインタ));
  
```

図2 APからの使用例

表1 「意紙」の基本API

・意紙の生成	・意紙のリンク
・意紙の削除	・意紙のアンリンク
・意紙のマップ	・意紙のリンク関係を得る
・意紙のリマップ	・意紙の属性を得る
・意紙のフラッシュ	・意紙の名前を変更する

6. 「意紙」管理機構の試作

V4 上で「意紙」管理機構の試作を行い、「意紙」によるデータ管理構造でオブジェクト間の整合性が保てるか確認した。また、試作では MS-DOS ファイルサーバを用いて MS-DOS のファイルシステムを利用することで「意紙」の永続化を行った。

6.1 「意紙」の実現

V4 はワンレベルストアを採用しており、「意紙」の実体はメモリオブジェクトにマッピングされる。そして図 2 で示すように管理テーブルを用意し、実体、属性、リンク構造、操作関数へのポインタを示す。操作関数とは「意紙」の操作時にダイナミックリンクされる関数である。

6.2 リンク管理部の実現

すべての「意紙」は管理テーブルを持っており、管

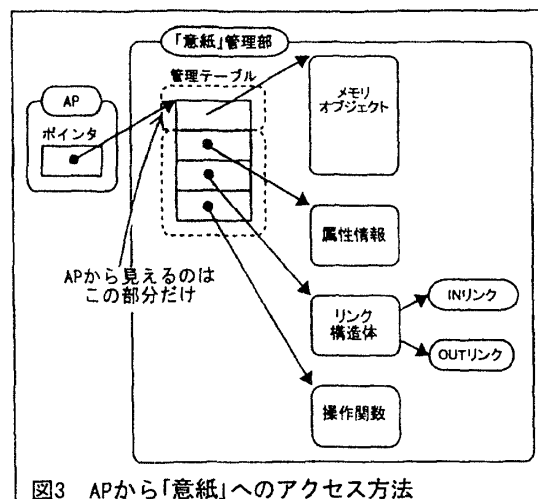


図3 APから「意紙」へのアクセス方法

理テーブルからその「意紙」のリンク関係を得ることができる。リンク関係はリンク元を示す IN リンクとリンク先を示す OUT リンクからなる。リンク管理部は「意紙」の操作に対して、リンク情報を更新することで、リンクの整合性を保つことができる。

6.3 属性管理部の実現

属性変換は属性ごとの操作関数で実現する。試作ではビットマップとして BMP, V4bitmap, テキストとしてフル 2 バイト JIS, シフト JIS といった属性を用意し、ビットマップ同士、テキスト同士での相互変換を実現した。

7. おわりに

本稿では、データ管理構造として「意紙」を定義し、その管理機構の試作を行った。「意紙」を提供することにより、ユーザのオブジェクト操作に対して、ユーザがオブジェクト間の整合性を取る必要がなくなった。また、オブジェクトに複数の属性を持たせることができるため、属性ごとの操作が可能になった。現段階では複数のタスクから「意紙」が参照されている場合、あるタスクで変更された「意紙」間の関係を他のタスクに通知する手段がない。今後はこのような「意紙」の同期処理を設計する予定である。

謝辞

本研究は、文部省科学研究費補助金（基盤研究（B）（2）課題番号 08458064）により行われた。

参考文献

- [1] S.J. Leffler, M.K. McKusick, M.J. Karels, J.S. Quarterman 共著, 「UNIX 4.3BSD の設計と実装」, 丸善, 1991