システムソフトウェアと
オペレーティング・システム 77－33
マルチメディア通信と分散処理 87－33
（1998．2．27）

OS/omicron 第4版のデータ管理機構 「意紙」サーバにおけるリンク機構の設計と実現

高野了成†，佐藤元信†，早川栄一†，並木美太郎†，高橋延匡‡
†東京農工大学　工学部　‡拓殖大学　工学部

〒184-8588　東京都小金井市中町2-24-16
E-mail：takano@omicron.ei.tuat.ac.jp

本稿では，OS/omicron 第4版のデータ管理機構である「意紙」サーバにおけるリンク機構の設計，実現について述べる．「意紙」はパターン情報など多様性，多義性，そして関連性を持つデータを表現するためのデータモデルである．従来のファイルシステムでは，アプリケーションが扱うデータ間のリンクには関与しないので，ファイルの名前空間の変更に対してリンクは整合性を保証することができなかった．そこで，「意紙」サーバにおいてリンク機構を実現し，システムで単一の名前空間に配置された「意紙」に対して，整合性が保証されたリンクを提供した．また，リンク機構を利用するための API を提供し，実際のアプリケーションに利用した．

Design and Implementation of a Link Mechanism in "ISHI" server on OS/omicron Version 4

TAKANO Ryousei †, SATO Motonobu †, HAYAKAWA Eiichi †,
NAMIKI Mitarou † and TAKAHASHI Nobumasa ‡
† Department of Computer Science, Tokyo University of Agriculture and Technology
‡ Department of Computer Science, Takushoku University

2-24-26 Nakacho Koganei, Tokyo, 184-8588 Japan
E-mail : takano@omicron.ei.tuat.ac.jp

This paper describes a design and implementation of a link mechanism in "ISHI" server on OS/omicron version 4. The charactaristics of pattern data are based on polysemy, diversity, and relation. Therefore we propose a data model called "ISHI" that supports these charactaristics. In UNIX filesystem, a link represented with relation between data can't guarantee consistency of relation. Because the filesystem doesn't share data structure in many applications. As a result, we design "ISHI" in system's unique namespace in "ISHI" server, and provide the consistency of link between instances of "ISHI". And we present the link API and apply to applications on OS/omicron Version 4.

1. はじめに

近年，計算機で扱うことができる情報が多種多様化してきたと同時に，それらの情報の関連を管理することが重要になっている．そこで，システムはユーザに対し，関連の容易な管理，記述性を提供することが要求される．

筆者らは，パターン処理を指向した OS/omicron 第 4 版（以下，V4）[1] の研究・開発を行っている．この V4 を研究基盤として，動画再生系，手書き入力による発想支援，グループウェアなどの研究が行われており，大量のパターン情報とその関連を管理する必要がある．

筆者は，パターン情報を扱うためのデータモデルとして「意紙」[2] を提案し，V4 のデータ管理機構として「意紙」サーバを実現した．本稿では，「意紙」サーバにおいて関連を管理するリンク機構の設計と実現について述べる．

2. データ管理おけるリンク機構

2.1 OS/omicron 第 4 版の特徴

V4 はパターン指向の OS であり，紙をメタファとした「電紙」をデータモデルとして提供する．「電紙」は多様な型を持つデータをモデル化し，データ型に応じた資源管理の拡張を容易にする．また，データ間の関係をポインタで表現し，リンク処理を高速化し，記述を容易にするためにワンレベルストアを採用している．V4 の全体構成を図 1 に示す．

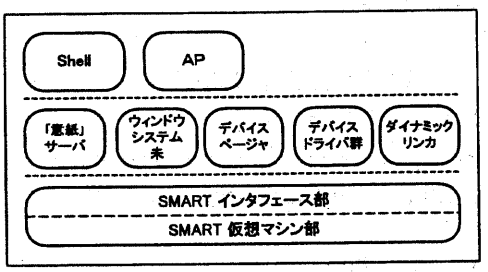

図 1. OS/omicron 第 4 版の全体構成

2.2 パターン情報の特徴

紙をメタファとすると，手書き情報など，パターン情報の特徴である多様性と多義性を扱うことができる必要がある．紙は文字や図形など，書く対象が制限されない．また，手書き情報は同一のストロークを入力した場合，それを文字としても，図形としても認識することができる．このように，多様性は一つの情報が複数の翻訳を持つという多義性を伴う可能性もある．

さらに，紙は自由に切貼りできるなど，柔軟な関連性を持っている．

2.3 従来のリンク管理方式

リンク対象と永続化の視点から，従来のシステムにおけるアプリケーション層，システム層におけるリンク管理について述べる．

(1) アプリケーション層

アプリケーションがデータ構造（データ間の関連）を表現するには，ワープロなどのようにアプリケーション独自の形式を用いてリンク情報を持つか，WWW ブラウザのように SGML,HTML などのプラットホームに依存しない（標準的な）構造記述言語を用いることで解決される．そして，このようなデータ構造は単なるバイト列に直列化 (serialization) され，ファイルとして永続化される．リンクはファイル単位とは限らず，ファイル内のある位置を参照することも可能である．

(2) システム層

ファイルは木構造の名前空間に配置される．UNIX 型のファイルシステムではハードリンク，シンボリックリンク [3] と呼ばれる二つのリンクを提供することで，名前空間を拡張している．

しかし，アプリケーション層のリンクとファイルの空間空間とは完全に分断されており，ファイルの移動，削除といった操作による空間空間の変更はリンクに反映されない．したがって，システムがリンクの整合性を保証することはできないので，アプリケーションがリンクを管理する必要がある．

このように，リンクには可搬性と整合性のトレードオフが存在する．

2.4 ファイルによる抽象化の限界

上述したように，UNIX 型のファイルシステムではデータの性質に関わらず，データをバイト列に抽象化し提供している．しかし，パターン情報は 2.2 で述べたように，構造や関連を持っており，これらの構造をそのままファイルに格納することはできない．したがって，パターン情報を効率的に扱うためにファイルに代わるデータモデルが必要である．

2.5 目的

紙を自由に切貼りする，といったオブジェクト操作をシステム層で実現する場合，システムがオブジェクト間の構造を管理する必要がある．

データ間の関連を管理するために提供するリンク機構の目的を次に示す．

・整合性が保証されたリンクをアプリケーションに提供する
・リンクはシステムが一意に管理し，すべてのアプリケーションから同一のインタフェースで利用できる

3.「意紙」の概念

「意紙」は「電紙」の特徴の中でも，オブジェクト間の関連性に注目したデータモデルであり，「意紙」間はネットワーク構造のリンクを持つ．また，「意紙」は複数のデータ型を属性として持つことが可能で，「意紙」の実体は各属性のインスタンスの集合と考えることができる．

3.1 関連性

「意紙」は紙のように，文章の書かれた紙の上に図表を書いた紙を貼りつけることができる．このようなオブジェクト間の関連性を「意紙」はリンクとして持ち，図 2 のように，木構造に限定されないネットワーク構造の関連を表現することができる．

また，「意紙」は，データ構造の永続化のために，従来のファイルシステムのようなファイルとディレクトリという二つのモデルを併用するのではなく，一つのモデルで実体と関連性を表す．

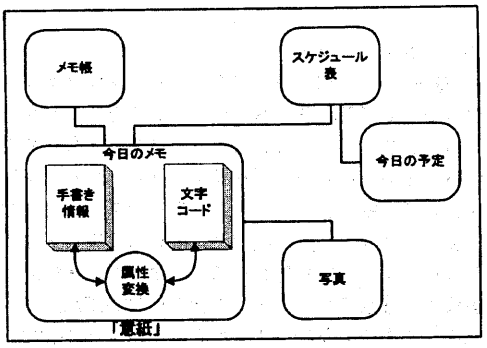

図 2.「意紙」とリンク

3.2 多態性

「意紙」はデータ型を表す属性を持っている．また，手書き情報が文字認識によって文字コードに変換できるように，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．「意紙」はこのように属性が異なる複数のオブジェクトを一つの論理的単位として扱うことができる．そして，各属性間で相互に変換できる属性変換機構を用意することで，「意紙」の操作に対する多態性を確保する．

3.3 拡張性

また，システムが有効な属性を規定してしまうと，属性を追加することが困難になり，多様な属性に対応することができない．そこで，継承を利用することで新しい属性を追加できる拡張性を提供する．

4. リンク機構の設計

4.1 オブジェクトの識別

オブジェクトを識別するために，仮想記憶のアドレスを使用する方法とオブジェクト表のインデックスなどの論理的な識別子を使用する方法がある．

アドレスを識別に用いた場合，言語処理系のポインタでオブジェクトを識別することができる．したがって，オブジェクト間の関連をポインタで表現することができる．

一方，オブジェクト表のインデックスを識別子として利用し，間接的にオブジェクトにアクセスすることで，オブジェクトに対するリンクの表現，アクセス制限，バージョン管理などが容易になる．たとえば，オブジェクトがバージョン番号を持てば，履歴管理機構［4］を利用することでオブジェクトの過去のバージョンを，システム全体のスナップショットからオブジェクト単位で参照することができる．

4.2 リンクのモデル

リンクはオブジェクトの削除や，移動に対してリンクのぶら下がり（dangling）が起こらないように整合性を保証する．そのためにオブジェクトがどのオブジェクトを参照しているかだけではなく，どのオブジェクトから参照されているかという双方向の参照関係を知る必要がある．

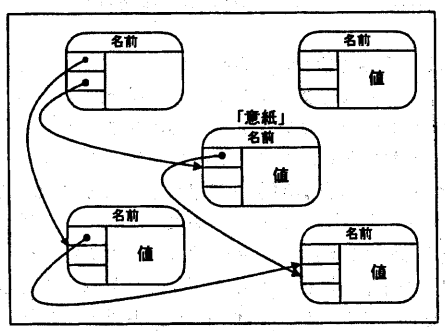

図 3.「意紙」のモデル

ポインタはオブジェクトを指し示すだけの一方向の関係だが，リンクはオブジェクトとオブジェクトの一対一，一対多，多対多の関係である．したがっ

て，単なるポインタではリンクを表現することができない．

「意紙」は，リンクの整合性，オブジェクト表の情報を利用することによる柔軟性を重視するため，オブジェクトの識別にはオブジェクト表のインデックスを識別子として用い，リンクによってオブジェクト単位の参照を表現する（図3）．

4.3 システム構成

V4 はマイクロカーネルアーキテクチャを採用しており，「意紙」の管理機構はマイクロカーネル SMART [5] 上のサーバとして実現する．この管理機構は2次記憶上のオブジェクトと識別子の対応を管理する Object Storage Server と名前空間，リンクの管理を行う「意紙」サーバから構成される．アプリケーションはランタイムライブラリとリンクされ，ランタイムライブラリは属性変換関数など，実行時に必要なライブラリとダイナミックリンクされる．全体構成を図4に示す．

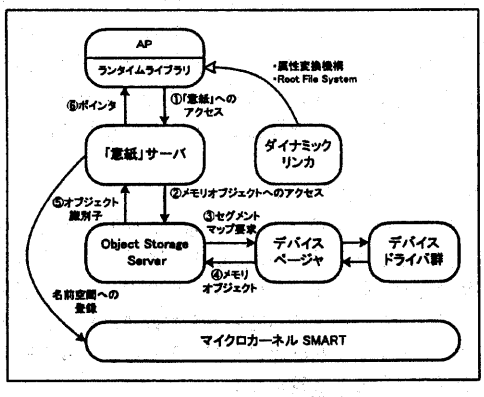

図4．モジュール構成

(1) Object Storage Server

V4 ではセグメントマッピング方式 [6] により，すべてのデバイスをメモリのインタフェースで扱うことができる．セグメントマッピング方式でメモリにマップした資源をメモリオブジェクトと呼び，2次記憶もデバイスページャによってマップされる．Object Storege Server はメモリオブジェクトやリンク情報が格納されたオブジェクト表を管理し，メモリオブジェクトとオブジェクト表のインデックスである識別子の対応をとる．

(2)「意紙」サーバ

「意紙」サーバはアクセス要求を受けるとアクセスが許可されているか調べ，Object Storage Server へ要求された属性のメモリオブジェクトのマップを要求する．また，Object Storage Server が提供するリンク情報を利用し，整合性が保証されたリンクをアプリケーションに提供する．

また，V4 ではすべての資源はシステムで単一の名前空間に配置される．図 5 は名前空間内の「意紙」の名前木を表示している．“意紙_ ROOT” 以下が「意紙」の名前空間である．「意紙」サーバは，その名前空間内に「意紙」の名前空間を対応付ける．したがって，システムで一意のリンクを提供することができる．また，「意紙」サーバの提供する名前空間とリンク機構は連動しているので，リンクの整合性は保証される．

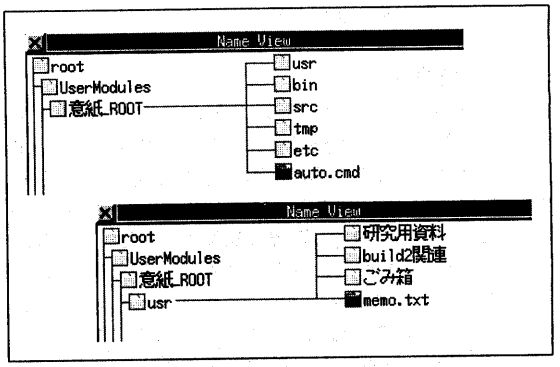

図5．名前空間

(3) ランタイムライブラリ

「意紙」サーバによってアプリケーションは実際のメモリオブジェクトへのポインタを得ることができる．アプリケーションはポインタをそのまま利用することも可能だが，通常はランタイムライブラリを介してメモリオブジェクトにアクセスする．ランタイムライブラリは，実行時に必要な関数をダイナミックリンクして実行される．たとえば，属性変換機構や，複数のファイルシステムに対して透過的に標準入出力関数を利用するための Root File System が存在する．このようにダイナミックリンクを利用することで拡張性を確保できる．

4.4 アプリケーションインタフェース

「意紙」サーバの提供する API における「意紙」の識別は名前によって行う．「意紙」の生成が要求された場合，「意紙」サーバは「意紙」，そしてその実体となるメモリオブジェクトを生成し，そのポインタを返す．

「意紙」サーバは表1に示すリンク関連 API を提供する．通常のリンクは，オブジェクト間の関連を示すだけある．しかし，実際利用する場合にはリンク自体に名前や属性を持たせたいことがある．そこ

で，名前付きのリンクを API として提供しているが，内部でリンク表現用のオブジェクトを生成しているだけで特別な処理はしていない．

表1. リンク関連 API

```
STATUS 名前付きの意紙リンクを生成する
                    ( CHAR *名前, 属性型 属性 );
STATUS 意紙リンクを張る( CHAR *リンク元, CHAR * リンク先 );
STATUS 意紙リンクを切る( CHAR *リンク元, CHAR *リンク先 );
オブジェクト構造体 *意紙リンクを得る
                    ( CHAR *名前, ULONG オフセット );
```

5. リンク機構の実現

5.1 実現環境と規模

これらの設計をもとに，PC/AT 互換機（Pentium 166MHz，メモリ 48MByte,EIDE 2GByte）上で動作する V4 上の「意紙」サーバにリンク機構を実現した．また，記述言語には当研究室で開発された言語 C 処理系である CAT386 を用いた．実現規模を表2に示す．

表2. 実現規模

「意紙」サーバ	1,500
Object Storage Server	3,100

5.2 「意紙」の内部構造

「意紙」は複数のメモリオブジェクトの集合であり，オブジェクト表はメモリオブジェクトの管理表である．V4 ではワンレベルストアを採用しており，メモリオブジェクトのアドレスはシステムで一意である．したがって，オブジェクト表をアプリケーションごとに持つ必要がなく，システムが一つの表を管理すればよい（図6）．

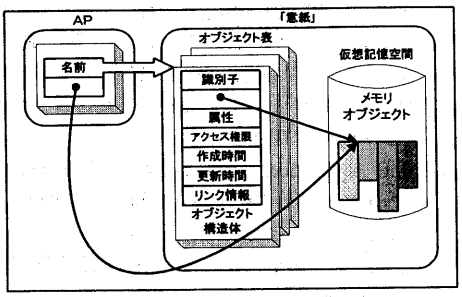

図6.「意紙」の内部構造

Object Storage Server はメモリオブジェクトごとに存在するオブジェクト構造体を管理する．オブジェクト構造体は UNIX ファイルシステムにおける inode のような作成時間，2 次記憶の構成ブロックなどの情報に加え，オブジェクト間のリンク情報を持つ．リンク情報にはリンク先すべてのオブジェクト識別子が格納されている．リンク情報を格納するために Object Storage Server は独自のフォーマットで 2 次記憶を管理する．

5.3 整合性を保証するリンク機構

「意紙」は管理の容易さから，木構造の名前空間とその空間に対するネットワーク構造のリンクを提供する．したがって，Object Storage Server はメモリオブジェクトに対して木構造の名前空間を提供し，その名前空間に対するネットワーク構造のリンクを「意紙」サーバが提供する．

「意紙」が削除された場合，その「意紙」のオブジェクト構造体に含まれているリンクをすべてたどり，リンク先で発生しているリンクのぶら下がりを削除する（図 7）．したがって，リンクの整合性が保証される．

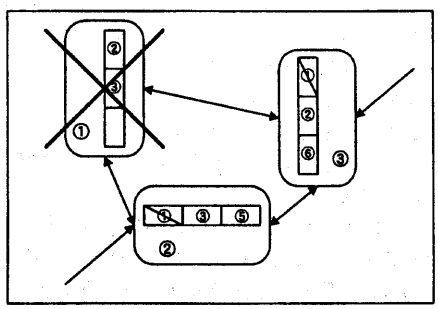

図7. リンクの整合性

5.4 アプリケーション例

「意紙」サーバのリンク機構を利用したアプリケーションである「Desktop 付箋紙」を作成した（図8）．

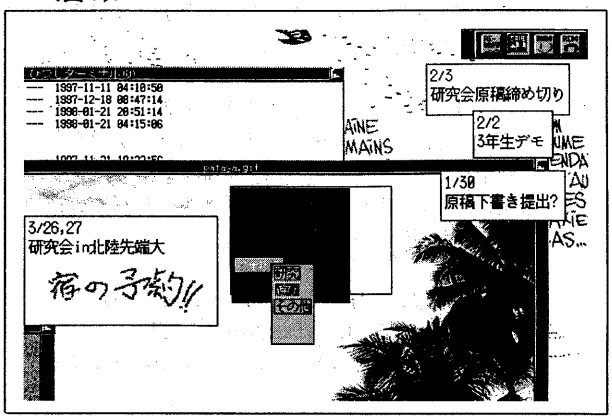

図8. 実行画面

この仮想的な付箋紙はテキストや手書き情報を書き込んだり，ビットマップを貼付けるができ，Desktop 画面の任意の位置に配置することができる．各付箋紙はカテゴリに分類されるが，リンクを利用することで，複数のカテゴリに属する付箋紙を関連付けることができる．

6. 関連研究

アプリケーションにおけるデータ構造を永続化するために，ファイルシステムを利用する場合は，ユーザが直列化の処理を書く必要がある．一方，データ構造をそのまま永続化するためにオブジェクト指向データベースを利用するというアプローチも考えられる．

ObjestStore [7] では，ランタイムライブラリをアプリケーションとリンクすることで，永続オブジェクトに対するアクセスを制御している．永続オブジェクトはアプリケーション内のヒープ領域に読み込まれ，オブジェクト指向言語から一時オブジェクトと透過にアクセスされる．永続オブジェクトの識別には仮想記憶のアドレスを利用し，オブジェクト間の参照をポインタで表現している．このため 2 次記憶に格納された参照をポインタとして利用可能にするために pointer swizzling [8] と呼ばれるポインタの変換が行われる．

V4 は OS レベルでワンレベルストアを提供しているので，「意紙」サーバはデータベース操作に限定しない汎用的なオブジェクトの永続化を提供することができる．

7. おわりに

本稿では，パターン情報を管理するためのデータモデルである「意紙」を提案し，OS/omicron 第 4 版のデータ管理機構である「意紙」サーバにおけるリンク機構の設計と実現について述べた．本研究の成果として次の点が挙げられる．

- アプリケーションに整合性が保証されたリンクを提供するために，「意紙」サーバのリンク機構を実現した
- リンク機構を利用するための API を提供し，実際のアプリケーションに利用した

また，今後の課題として，次の点が挙げられる．

- 「意紙」における属性の拡張機構の実現
- 言語 C++ 処理系を利用したクラスライブラリの実現

謝辞

本研究の一部は，文部省科学研究費補助金（基盤研究(A)(2)09358004，基盤研究(B)(2)08458064，奨励研究（A）09780255）により行われた．

参考文献

[1] Eiichi Hayakawa, Mitarou Namiki, Nobumasa Takahashi : "Basic Design of SHOSHI Operating System that supperts Handwriting Interfaces", IPSJ, Vol. 35, No. 12, pp. 2590-2601, 1994

[2] 高野了成，佐藤元信，早川栄一，並木美太郎，高橋延匡："OS/omicron 第 4 版におけるデータ管理機構の設計と実現"，電子情報通信学会研究報告，Vol. 97, No. 87, pp. 13-18, 1997

[3] M. K. McKusick, K. Bostic, M. J. Karels, J. S. Quarterman : "The Design and Implementation of the 4. 4BSD UNIX Operating System", Addison-Wesley, 1996

[4] 横田大輔，高野了成，早川栄一，並木美太郎，高橋延匡："ワンレベルストア上での開発を指向した履歴管理機構の設計と実現"，情報処理学会第 8 回コンピュータシステムシンポジウム論文集，Vol. 97, No. 8, 1997

[5] Tomoyuki Morinaga, Motonobu Sato, Eiichi Hayakawa, Mitarou Namiki, Nobumasa Takahashi : "Throw : An Efficient and Extensible Structure Model for Microkernel Architecture", In Proceedings of the IPSJ International Symposium on Information Systems and Technologies for Network Society, pp. 133-140, 1997

[6] 佐藤元信，森永智之，早川栄一，並木美太郎，高橋延匡："OS/omicron 第 4 版のデバイス管理におけるシーケンシャルデバイス管理法式の拡張"，情報処理学会第 7 回コンピュータシステムシンポジウム論文集，Vol. 96, No. 7, pp. 179-186, 1996

[7] Object Disign, Inc. : "ObjectStore Technical Overview", 1997

[8] J. Eliot B. Moss : "Working with Persistent Objects : To Swizzle or Not to Swizzle", IEEE Transactions on Software Engineering, vol. 18, No. 8, pp. 657-673, 1992