システムソフトウェアと
オペレーティング・システム 84−16
（２０００．５．２５）

メディア変換機構を提供する永続オブジェクトシステムの設計と実現

高野了成†，佐藤元信†，早川栄一‡，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@os-omicron.org

分散環境において協調作業を行う場合，データ間の参照関係などの構造を共有し，一貫性を保証する必要がある．さらに，各データは利用環境や共有相手の環境に応じて，フォーマット変換，パターン認識などの変換処理が実行され，利用される．我々は，このような作業を capture-arrange-publish の３フェーズから成るモデルとして定義し，メディア変換機構を提供する永続オブジェクトシステムを設計，OS/omicron 第４版上への実装を行った．本システムの特徴は，(1) オブジェクト間の関連性をポインタで記述するために，オブジェクト位置に透過なオブジェクト空間管理を提供すること，(2) メディア変換を効率的に行うためのキャッシュとその一貫性管理を提供すること，(3) メディア変換時の対話的処理を支援するためのスクリプト言語の呼出し機構を提供することである．

Design and Implementation of a Persistent Object System with Media Transcoding Mechanism

TAKANO Ryousei †, SATO Motonobu †, HAYAKAWA Eiichi ‡,
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@os-omicron.org

An object management system ensures consistency of sharing complex data structures that contains several relations in cooperative works on distributed computers. Furthermore, data processing requires context-dependent operations : format convertion, pattern recognition and so on. We define a working model to consist of 3 phases : capture-arrange-publish. We design a persistent object system with media transcoding mechanism, and implement on OS/omicron V4. Features of this system are (1) location transparent object space management to describe relations of data by pointer, (2) object cache and consistency management to support efficient media transcoding, and (3) calling mechanism of scripting languages to support user interaction process at media transcoding.

1. はじめに

近年，計算機を文書作成や，研究開発などの発想活動支援に利用する機会が増加している．個人がPDA などの携帯端末を含めた性質の異なる複数の計算機を所有し，利用環境に応じて使い分けたり，そこで生まれたデータを複数人でも共有し，協調作業を行うことが多い．このような環境下では，データは互いに密な関連性を持ち，利用されるマシン環境やアプリケーションごとに多様であり，元が同じデータでも利用環境や共有相手の環境に応じてデータ表現を変換することが必要になってくる．さらに，データを整理し，活用する場合には，入力されたデータを元に，別の計算機上で認識や抽出などの処理を行うことで，データの抽象度を上げていく必要がある．そこで，不定型で断片的なアイデアを逃さず記録し，参照関係，順序関係などの構造を保持したまま共有空間へ格納でき，加工，編集しやすいデータ形式に変換する機構を提供するフレームワークが求められている．

我々のプロジェクトでも，ペンインタフェースによるグループウェアである分散手書き KJ 法 [3] や，複数人によるソフトウェア開発を支援する電子研究ノート [4] などの研究を行って来た．これらのアプリケーションでは，発想の記録を重視して，まずは，データ自体をビットマップや，ストロークといった一次情報で格納し，後にコード情報へと変換することを想定している．このような環境では，一つのデータが複数のデータ表現を持つことになり，これらを効率的に提供するデータ管理機構が必要となってきた．

このような要求に対して，我々はメディア変換機構を特徴とする永続オブジェクトモデルである「意紙」[1] を用いたデータ管理システムを提案している．「意紙」は一つのデータに対する複数の表現形式を抽象化し，分散透過な永続オブジェクトを提供する．

本稿では，まずターゲットとする作業のモデル，システム規模について述べ，本システムの目的を示す．次に「意紙」に基づく永続オブジェクトシステムの設計について述べ，OS/omicron 第 4 版上への実装方式について述べる．

2. 要求分析

本章では，本システムのターゲットである発想活動支援における計算機利用の作業モデルを定義し，システムに対する要求分析について述べる．

2.1 作業モデル

計算機において知識を効率的に蓄積，編集，再利用することが重要であるという視点から，我々は capture-arrage-publish [2] の 3 フェーズから成る作業モデルを提案した．次に各フェーズの作業の概要を示す．

(1) capture

さまざまなフォーマットのデータや，データ取得に伴う時間情報，位置情報など任意の属性情報を単一のデータエントリに格納する．

(2) arrange

属性情報を利用した検索やフィルタリングを行ったり，データ間のリンクを操作して，データを加工する．

(3) publish

WWW や紙への印刷物（出力）として，または他のモジュールへデータを出力する．

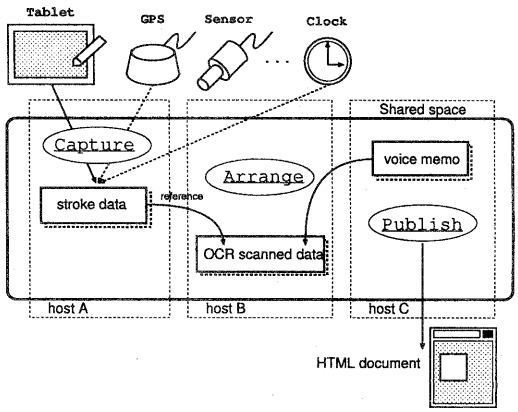

図 1： 作業モデル

例えば，HTML 文書は閲覧することは容易であるという点で出力形式としては問題ないが，リンクが単方向の参照指向な構造であり，変更に対する整合性や保守性が考慮されていないという点や，入力方式がフォームによるテキストに限られ，手書きストロークなどの多様な情報が扱えない点など，編集に適したフォーマットではない．図 1 に示した例では，OCR で取り込んだ文書に対して，タブレットから手書きストロークによるメモを書き込んでいる．手書きストロークには位置情報，時刻などの属性情報を付加して保存する（capture）．そして，capture した各データを属性情報を利用してフィルタリングし，同時刻，同位置で取得した音声メモに対しても関連付けを行う（arrange），最後に，このデータを HTML 文書に変換して，WWW 上で公開する（publish）．

2.2 メディア変換

分散環境ではデータに対して (1) 個人での共有性と (2) 他者との共有性が生じる．(1) は PDA

とデスクトップ PC など性質の異なる複数の計算機で同一のデータを扱う点，(2) はデータフォーマットの可搬性が必要になる点が問題になる．そこで，各フェーズで適切な操作が行えるように，データ共有に伴う変換処理を一貫して扱う枠組みをシステムが提供することが要求される．なお，データの表現形式を変換することをメディア変換と呼び，各表現形式をメディア型と呼ぶ．次にメディア変換の例を挙げる．

・画像，音声のフォーマット変換
・文字認識，OCR（Optical Character Reader）
・仮名漢字変換
・バージョン管理

メディア変換には可逆性，インタラクティブ性の有無などの違いがある．そこで，変換によってデータの欠損が起きることを通知したり，変換に伴うユーザとの対話処理を支援する必要がある．

2.3 システムへの要求

上記のような作業を行うためのシステムに求められる要求を次にまとめる．

(1) データ共有が容易な分散永続オブジェクト機構

記憶階層や分散環境に対して透過的にオブジェクトを操作したい．さらに，データ間には密な関連性があるので，このような関連性をプログラミング言語から容易に操作したい．

(2) 拡張可能なメディア変換機構

メディア型の種類はシステムがあらかじめ決定することができないので，拡張可能な機構が必要である．また，メディア型によってアクセスパターンが異なるので，オブジェクトごとにポリシの選択ができるようにしたい．

(3) オブジェクト指向スクリプト言語

認識処理を伴うメディア変換などでは，変換結果が静的に決定できない可能性があり，ユーザとの対話処理が必要になる．このようにシステムであらかじめ指定することが困難なデータの抽出や加工などの操作は，ユーザが記述したスクリプトにより処理できるようにしたい．

2.4 システムの規模

本システムは個人および大学の研究室規模の協同プロジェクトで利用することを前提としており，想定するシステムの規模は次のようになる．

・10 ～ 20 台程度のネットワーク接続された PC
・同時に共有するオブジェクトは数百個程度
・オブジェクトサイズは数百 k ～数十 MB

3. 目的

本章では，前章で述べた要求を解決するためにシステムが実現する目的について述べる．

3.1 多態的オブジェクトの提供

マルチメディアデータなど多態的な性質を持つデータは，一つの実体に対して認識処理やフォーマットの違いから複数の表現形式を持っていると言えるが，このような複数のメディア型のオブジェクトを一つの集合として扱えるようにする．このような多態的オブジェクトを「意紙」と呼び，「意紙」に対してメディア変換を行うことで，要求するメディア型のオブジェクトを得ることができる．

また，システムで統一的にメディア型を管理し，動的にメディア型を追加，削除できるようにする．

3.2 分散永続オブジェクトの提供

オブジェクト間には，依存関係や親子関係などさまざまな関連性が存在し，各オブジェクトは分散環境上に散在している．このような関連性をポインタを含む構造として扱う場合，ポインタの永続性やリモート計算機上のポインタの扱いが問題になる．そこで，ポインタの永続性やリモート計算機上のポインタをローカル計算機上のポインタと透過的に扱える分散永続オブジェクトを提供する．

また，複数言語を利用して分散永続オブジェクトを共有するためには，言語処理系のランタイムにおいてオブジェクトを抽象化し，一意性を保証するのではなく，言語非依存なアドレス空間レベルでの共有機構を提供する．

4. 設計方針

本システムでは，それぞれのメディア型を持った分散永続オブジェクトと，その集合である多態的オブジェクトを提供する．本章では，これらの設計方針について述べる．

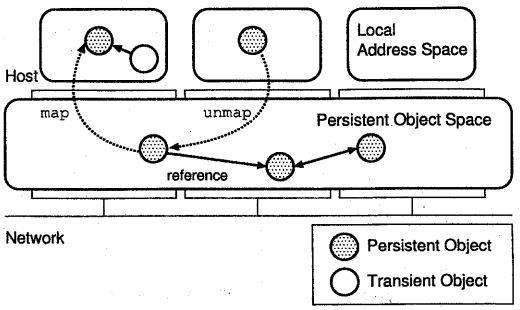

図2：アドレス空間モデル

4.1 アドレス空間モデル

アドレス空間モデルを図2に示す．永続オブジェクト空間上のオブジェクトは，永続かつ全システムでの一意性が保証されている．永続オブジェクトへのアクセスが発生した場合，永続オブジェクトがローカル計算機上の仮想アドレス空間にマップされ，アクセス可能になる．このように全マシンの仮想アドレス空間を共有，永続化するのではなく，永続オブジェクトに対してだけアドレス空間を共有するのは，一貫性を保持するコストを軽減するためである．なお，ローカル計算機の仮想アドレス空間へマップされた永続オブジェクトの複製をキャッシュオブジェクトと呼ぶ．

すべてのオブジェクトは永続的であるか，リモート計算機に存在するかに関わらず透過に扱えるようにする．このアドレス空間はシステムレベルで提供するので，言語非依存であり，永続オブジェクト間の参照をポインタとして記述することができる．したがって，オブジェクト間の関連性の記述が容易になる．ただし，永続オブジェクトから一時オブジェクトに対するポインタ値をシステムは保証しない．

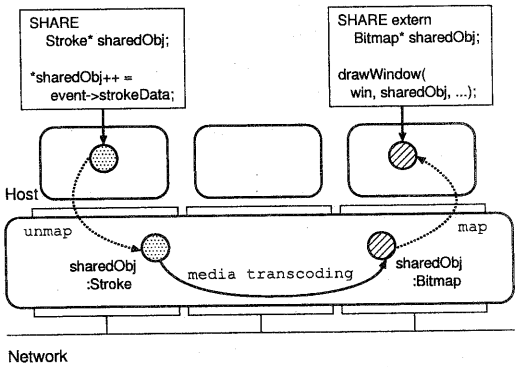

図3a：メディア変換機構（1）

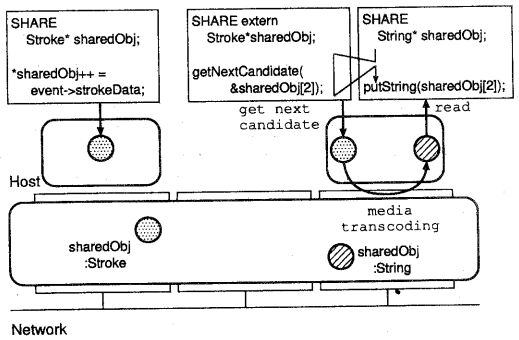

図3b：メディア変換機構（2）

4.2 メディア変換機構

アプリケーションから永続オブジェクトは識別子名とメディア型の組で参照することができる．図3a はストロークからビットマップというメディア変換にユーザとの対話処理が伴わない場合，図 3b はストロークから文字列という対話処理が必要な場合のメディア変換機構の処理を示している．図 3b の場合はストロークに対するメディア変換（文字認識変換）の結果，3 番目の文字が誤っていたため次候補を得ようとしている処理を示している．

仮想アドレス空間へのマップは，永続オブジェクトのキャッシングと等価である．メディア変換時にリモート計算機上の永続オブジェクトへのアクセスが発生したとしても，キャッシュが存在する場合はネットワーク通信が発生しない．したがって，ユーザとの対話的処理を行う場合は，暗黙的にキャッシュを利用することになるので効率的である．例えば，図 3b の場合はストロークデータへのアクセスにキャッシュを利用している．

前述したように，多態的オブジェクトは複数の永続オブジェクトから構成される．あるメディア型のオブジェクトを変更した場合，他のメディア型のオブジェクトも変更が必要になる可能性がある．このような影響の波及はアプリケーション依存であり，システムで自動化することが困難である．そこで，システムは同期プリミティブを提供し，プログラマが明示的に同期処理を記述できるようにする．

5. 設計

本章では，本システムの設計について述べる．設計方針を満たすために，本システムでは次の三つの問題点を解決する必要がある．

- 分散オブジェクトへの参照の効率的な検出
- 分散オブジェクトに対する参照を通常のポインタと透過に扱う方法
- 分散オブジェクト間の一貫性管理

5.1 分散オブジェクト管理機構

分散オブジェクトを共有するには，主に次の2種類の方法が考えられ，それぞれオブジェクトの一貫性を保持するためのプロトコルを提供しており，プログラムの記述性と性能はトレードオフの関係にある．

(1) 1次元アドレス空間を共有する
(2) 特定の共有変数だけを共有する

(1) は MMU 機構を用いることで，ページングをネットワークを介して行う．プログラマが分散を意識する必要がないので，プログラミングが容易であり実装も単純だが，ページの一貫性を確保するために制限の強い順序一貫性を適用する必要があり性能が悪い．一方，(2) はエントリ一貫性やリリース一

貫性など，一貫性の条件を緩和することができ，性能をチューニングすることができる．しかし，プログラマは共有変数の宣言に対して注釈を付加したり，臨界領域を指定する必要がある．

本システムでは，従来の分散共有メモリシステムのように大規模計算などの効率化を目指しているわけではなく，ユーザとの対話的処理に利用されるオブジェクトの共有を目的としている．したがって，共有したい分散オブジェクトの一貫性だけを管理すればよいことになる．

そこで共有する分散オブジェクトについては一意のアドレスを割り当てることとし，オブジェクトは変数名と対応付ける．リモート計算機に存在する永続オブジェクトに対してアクセスした場合は，永続オブジェクトの所有者を検索し，ローカル計算機の仮想アドレス空間にマップする．なお，所有者とは永続オブジェクトが存在する計算機のことである．

また，複数タスクが同時にあるオブジェクトに対して読み込むことは許すが，書き込みを許すタスクはシステムで一つとする．書込みが発生した時点ですべての複製を無効化する方法と，変更する方法があるが，ユーザが選択できるようにする．

5.2 メディア変換機構

オブジェクトへの参照は変数を利用すると述べたが，変数の型をメディア型の指定に利用する．指定したメディア型の有効なオブジェクトが見つかればアクセスに成功するが，存在しない場合は，メディア変換を利用して既存のオブジェクトから新規にオブジェクトを生成する．メディア変換機構は変換元メディア型，変換先メディア型，変換メソッド，変換メソッドに対する可逆可能性などのヒント情報を管理する．

メディア変換は単純にシーケンシャルな順序で処理できるわけではない．例えば，文字認識結果の候補から正しい文字を選択したり，画像の明彩度などのパラメータを指定するなどユーザとの対話処理が必要な場合，変換メソッド内で前もって指定することは不可能なので，ユーザが記述したスクリプトを変換メソッドから呼び出せるようにする．つまりネイティブコードとスクリプト言語をダイナミックリンクする機構を提供する．

また，メディア変換に伴う各オブジェクト間の一貫性を保持するために，システムがロック変数による同期を提供する．

5.3 提供するインタフェース

分散オブジェクト機構が提供する主なインタフェースを図 4 に示す．(a) はロック変数に対するインタフェース，(b) はメディア変換に対するインタフェース，(c) は分散オブジェクト機構内部で分散オブジェクトを共有するために利用するインタフェースである．

本システムではオブジェクト識別子として仮想アドレスを使用し，メディア型はコンパイラが出力する型情報をエンコードした文字列を使用する．

```
LOCK   dom_create_lock(VOID);
STATUS dom_delete_lock(LOCK lock);
STATUS dom_lock(LOCK lock, TIME timeout);
STATUS dom_unlock(LOCK lock);
```

(a) lock variable operation

```
STATUS ishi_register_transcoder(CHAR* type,
                                CHAR* func, ULONG hint);
STATUS ishi_release_transcoder(CHAR* type, CHAR* func);
```

(b) media transcoding operation

```
STATUS dom_register_object(OID id, CHAR* name, CHAR*type);
STATUS dom_release_object(OID id);

STATUS dom_notify(OID id, UWORD status, CHAR* name, CHAR* type);
       status = DSS_REGISTER|DSS_RELEASE|DSS_INVALIDATE

STATUS dom_write_page(OID id, IPADDR ip,
                      ULONG offset, ULONG size, UBYTE* ptr);
STATUS dom_read_page(OID id, IPADDR ip,
                     ULONG offset, ULONG* size, UBYTE* ptr);
```

(c) distributed object operation

図 4： 提供する主な API

6. 実現

本機構を PC/AT 互換機で動作する OS/omicron 第 4 版（以下，V4）上に実装した．ネットワーク環境は 10BaseT であり，NIC として 3Com 3C509,3C589 PCMICA カードを用いた．通信プロトコルとして TCP/IP を使用するが，プロトコルスタックは Comer らの実装 [12] を移植した．本章では，V4 の概要と V4 が提供する機構を用いた分散永続オブジェクトシステムの実装について述べる．

6.1 OS/omicron 第 4 版の概要

V4 のシステム構成を図 5 に示す．V4 はマイクロカーネル構成の OS であり，SMART マイクロカーネル上に OS パーソナリティ，ダイナミックリンカ，メモリ管理システムなどのシステムモジュールが存在する．そして，すべてのデバイスなどの資源はメモリ管理システムによってさまざまな属性を持つオブジェクトとしてメモリにマッピングされる．アプリケーションがこれらのオブジェクトにアクセスする場合，ダイナミックリンカが変数の属性，識別子名，実際のアドレスなどの情報を含むリンケージテーブル経由で名前空間を検索することで動的にリンクを確定する．

なお，アドレス空間はセグメント ID (32 ビット)，オフセット (32 ビット) の 64 ビットで表現される単一 2 次元アドレス空間を採用している．

6.2 システム構成

分散オブジェクト機構の構成を図6に示し，各要素について次に述べる．

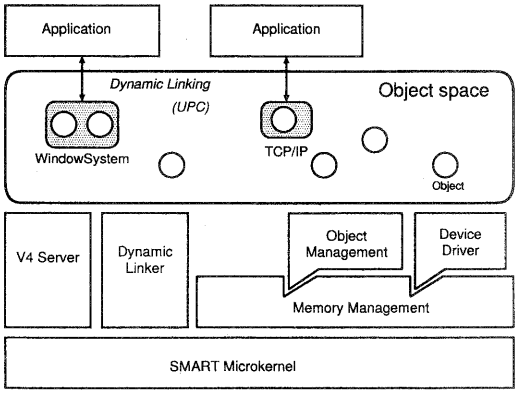

図5： OS/omicron 第4版の全体構成

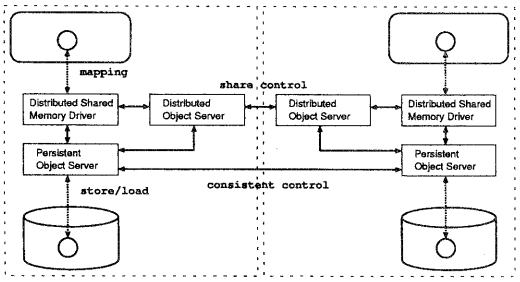

図6： 分散オブジェクト機構

(1) 分散共有メモリドライバ

分散共有メモリドライバは，メモリ管理機構のインスタンス(Machの外部ページャに近い)として実装され，分散オブジェクトへのアクセスに対するセグメンテーションフォールト，ページフォールトなどのバッキングストア操作を定義している．

例えば通信粒度をページ/セグメント単位にするなど，メディア型のアクセスパターンに応じたプロトコルのドライバを作成し，アプリケーションが永続オブジェクトの種類に適切なドライバを選択することが可能である．

(2) 分散オブジェクトサーバ

各マシンの分散オブジェクトサーバと協調して，オブジェクト識別子を管理することで，分散オブジェクトの一意性を保証する．また，分散オブジェクトの所有者，キャッシュオブジェクトの所有者，ロック変数を管理する．

また，永続オブジェクトにおける名前空間の同期も管理する．名前空間は共有変数名とメディア属性から永続オブジェクトを検索するために利用する．

(3) 永続オブジェクトサーバ

永続オブジェクトやそのキャッシュオブジェクトに対する2次記憶上のバッキングストアを管理する．また，後述するポインタ変換の管理も行う．分散環境に特化しているわけではなく，スタンドアローン環境でも同様に動作する．

6.3 共有変数と永続オブジェクトの対応

永続オブジェクトは言語上からは共有変数として操作できる．V4では，変数などの識別子はダイナミックリンク時に，リンケージテーブル経由でアドレスに変換される．分散オブジェクトに対しても，この機構を利用し，共有変数のバインディングを遅延評価することで，実行時にオブジェクト所有者を検索，キャッシュし，アクセス可能な状態まで準備することができる．さらに共有変数は一意の永続オブジェクトと対応付けられるので，アプリケーションはオブジェクトを位置透過に扱えるようになる．また，オブジェクトの所有者が変わっても，プログラムを変更する必要はない．

しかし，すべての大域変数を分散共有するわけではないので，リンケージテーブルの属性として共有属性を追加する．これで，リンケージフォールト時に共有するのかどうかの指定が可能になる．共有変数に対する名前空間の管理は各計算機上の分散オブジェクトサーバが行う．

ダイナミックリンクを利用した変数と永続オブジェクトのバインディングについて図7に示し，永続オブジェクトに対するアクセス処理の流れを次に示す．

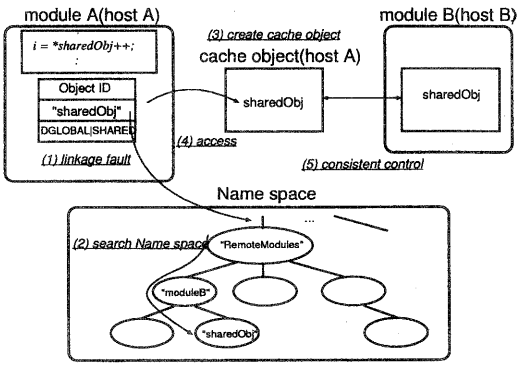

図7： 共有変数と永続オブジェクトのバインディング

なお，分散共有メモリドライバは転送粒度をセグメント単位として実装されているとする．セグメント単位では無効化のコストが高いので，書込みが大半を占める操作の場合は転送粒度をページ単位にするとよい．

(1) 最初に共有変数に対する参照が発生した場合，リンケージフォールトが発生する．

(2) ダイナミックリンカは名前空間を探索し，永続オブジェクトを特定する．存在しない場合は，新規に永続オブジェクトを生成する．

(3a) 特定した永続オブジェクトを参照すると，分散共有メモリドライバのマップハンドラが呼ばれ，永続オブジェクトサーバによって，ローカル計算機に永続オブジェクトのキャッシュオブジェクトが生成され，仮想アドレス空間にマッピングされる．

(3b) 分散共有メモリドライバは永続オブジェクトをネットワーク経由で転送し，キャッシュオブジェクトに書き込む．さらにキャッシュオブジェクトへの書込みを監視するため，セグメントを書込み禁止に設定する．

(4) キャッシュオブジェクトへの書込みによりセグメンテーションフォールトが発生すると，分散共有メモリドライバのセグメンテーションフォールトハンドラが呼ばれ，永続オブジェクトの所有者をローカルマシンに変更するように，分散オブジェクトサーバに要求を出す．

(5) 前所有者はキャッシュオブジェクトを持っているすべての計算機に対してキャッシュオブジェクトの無効化を通知する．

6.4 ネットワーク通信機構

分散セグメントサーバ間における制御通信はTCP を，ページ転送には UDP による通信を行う．制御通信には次のようなものがある．

・共有変数の登録，抹消
・キャッシュオブジェクトの無効化
・所有者の変更
・ページの転送要求
・ロック変数の確保，解放

6.5 ポインタ変換機構

理論的には 32 ビットのセグメント ID で表現できるセグメント数は 4G 個であるが，x86 プロセッサ上の制限により 8192 個しか利用できない．そこで，PointerSwizzling [6] の技術を利用してポインタ変換を行うことで，セグメント ID として 32 ビットの表現を利用できるようにした [1]．ポインタ変換に要するオーバヘッドはフォールト処理時間の 25%である．

6.6 スクリプトの呼出し機構

メディア変換における対話処理にスクリプト言語を使う場合，ネイティブコードからスクリプト言語，またその逆の呼出しを実現する必要がある．V4 ではダイナミックリンク機構を利用して，例えば Java などのインタプリタ型言語とネイティブコード間のリンクをプログラマが特別な手続きを書く必要なく実現することができる [5]．

この機構を利用することで，スクリプト言語の属性が設定されたコード領域にインストラクションポインタが来たときにインタプリタを実行することができる．また，図 8 に示すように，異なる言語モジュール間でダイナミックリンクが発生した場合，型変換やスタック操作など言語間の差異を吸収するためのスタブコードを挿入することができる．

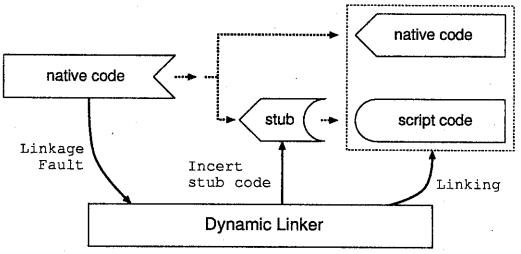

図8： ネイティブコードとスクリプト言語のリンク処理

7. 関連研究

Lucas [7] は，メモリマップ技術を分散環境に適応したリージョンという仮想記憶管理機構を採用しており，リージョンに対して，アクセスパターンに応じた一貫性プロトコルを提供している．リージョンは本システムにおけるセグメントとほぼ等価な概念であるが，セグメントは識別子が一意であり，アドレスの衝突が起きないため，再配置のオーバヘッドが軽減できる．また，リージョンの位置依存性，バージョン化，複合リージョンなどの関係を宣言するためにリージョン間参照を提供しているのに対して，本システムではメディア属性による複合関係を提供していると言える．

投機的処理を支援する世界 OS [8] は，並列世界モデルに基づいてファイル名と世界識別子からなる 2 次元名前空間を提供している．本システムにおける多態的オブジェクトはオブジェクト名と属性の 2 次元で表現でき，投機的処理もユーザの要求に先だってキャッシュを用意するメディア変換の一例と

考えることもできる．また，各世界の変更に対する伝播として継承（非対称），対称的，独立の三つのセマンティクスがシステムによって提供されるている．一方，本システムではメディア型に応じた分散共有メモリドライバを利用することで，アプリケーション依存であるオブジェクト間の一貫性管理を行うことが可能である．

ALAN [9] や Media Gateway [10] は，ネットワークの中間ノードにおいて画像，音声変換などのメディア変換を実行する．ALAN を利用した DPS (Dynamic Proxy Server) は，ユーザが記述した Proxylet をネットワーク経由で追加することによって，動的な機能拡張を可能にしている．しかし，Proxylet 間をストリームでデータ転送することを前提に設計されており，複数タスクによってオブジェクトを共有し，協調操作することは考えられていない．ANTS [11] は，新しいプロトコルを動的に配布し，広域分散しているネットワークインフラに対して段階的に展開することを目指しており，デマンドコード配送機構を提供することが特徴の一つとして挙げられる．データの共有だけでなく，コード（共有ライブラリ）を転送するために，本システムでもリンケージフォールトをトリガとすることで，適用可能な技術である．

8. まとめ

本稿では，メディア変換機構を提供する永続オブジェクトシステムの設計と，V4 における実装について述べた．本システムの特徴は，(1) オブジェクト間の関連性をポインタで記述するために，オブジェクト位置に透過なオブジェクト空間管理を提供すること，(2) メディア変換を効率的に行うためのキャッシュとその一貫性管理を提供すること，(3) メディア変換時の対話的処理を支援するためのスクリプト言語の呼出し機構を提供することである．

今後の課題として，メディア変換に利用するためのスクリプト言語を，既存の言語を移植するか新規に作成し，実用的アプリケーションによる評価を行うことが挙げられる．また，V4 だけで構成される単一なシステムではなく，他のシステムが混在する環境での相互運用性を実現することで，さまざまなメディア変換を利用できる環境を構築する．

参考文献

[1] 高野了成，佐藤元信，早川栄一，並木美太郎，高橋延匡： 多態的表現を可能にする永続オブジェクト管理機構，情報処理学会研究会報告，99-OS-81, pp.1-6, 1999.

[2] Eiichi HAYAKAWA, Tomoyo SATO, Ryousei TAKANO, Motonobu SATO, Nobumasa TAKAHASHI: Flexible, Modular System Architecture for Supporting Creative Work, In Adjunct Conference Proceeedings of HCI International'99, pp.163-164, 1999.

[3] 中島一彰，早川栄一，並木美太郎，高橋延匡：分散環境における発想支援のためのリアルタイム手書き協調作業システムの設計と実現，情報処理学会論文誌，Vol.38, No.12, 1997.

[4] 佐藤友代，並木美太郎，早川栄一：ソフトウェア開発過程の記録を支援する電子研究ノートの設計と実現，情報処理学会研究会報告，99-HI-83, pp.49-54, 1999.

[5] 山本康弘，早川栄一，並木美太郎，高橋延匡：OS/omicron V4 におけるインタプリタ型言語とネイティブコードのリンクの実現，情報処理学会第54回全国大会予稿集，pp.51-52, 1997.

[6] 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.

[7] 猪原茂和，上原敬太郎，宮澤元，益田隆司：オペレーティングシステム Lucas における 64 ビットアドレス空間の管理，情報処理学会研究会報告，93-OS-61, pp.81-88, 1993.

[8] 新城靖，孫軍： 並列世界モデルに基づくオペレーティング・システム，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，pp.95-100, 1997.

[9] Michael Fry and Atanu Ghosh: Application Level Active Networking, In Proceedings of HIPPARCH'98, 1998.

[10] Elan Amir, Steven McCanne, and Randy Katz: An Active Service Framework and its Application to Read-time Multimedia Transcoding, In Proceedings of ACM SIGCOMM'98, 1998.

[11] David L. Wetbcrall, John V. Guttag, and David L. Tennenhouse: ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols, In Proceedings of IEEE OPENARCH'98, 1998.

[12] Douglas E. Comer, David L. Stevens: Internetworking With TCP/IP Vol II: Design, Implemantation, and Internals, Prentice Hall, 1991.