データベースシステム　115－13
情　報　学　基　礎　49－13
（１９９８．５．１５）

ハイパーリンク型情報管理システムのＷＥＢ化およびＣＯＲＢＡ化の方式検討

窪野　哲光

東京電力株式会社 技術開発本部 システム研究所 マルチメディアデータベースＧ
〒２３０－８５１０　横浜市鶴見区江ヶ崎町４－１

文書情報へのアクセスに注目し，文書ファイル資源をオブジェクト指向ファイルシステム化し，それに対するハイパーメディア操作が出来るＧＵＩとを協調連係する「Ｗｒａｐｐｅｒ情報モデル」に基づくハイパーリンク型情報管理システムの開発を，動的オブジェクト指向言語Smaltalkにより進めてきた．メディア情報とリンク情報との陽な分離，ハイパーリンクの一貫性管理という特徴を有する本方式をネットワーク環境下に拡大するためのＷＥＢ化およびＣＯＲＢＡ化の方式検討と，開発言語の動的性を活かしたソフトウェアエージェントによる大域検索方式について述べる．

A Study on Web-zation and CORBA-zation of Hyperlink typed Information Management System Using Dynamic OOPL of Smalltalk

Norimitsu KUBONO

MulitiMedia DataBase Group　Computer And Communications R&D Center
Engineering Research And Development Division
Tokyo Electric Power CO.,LTD.
4-1, Egasaki-Cho, Tsurumi-Ku, Yokohama, 230-8510

We developed the Hyperlink typed information management system based on the original software platform"Wrapper information model" which consists of object oriented file system encapsulating file system with user-defined-extended profile and hypermedia interface.It realized an explicit separation of media and hyperlink, also integrity constraints for hyperlink. Recently, it is expected to apply Web technology and distributed object technology CORBA(Common Obejct Requst Broker Architecture & Specification) to a legacy system. So, with the aim of Web-zation and CORBA-zation of legacy system, we researched both plototype systems. We show that the essence of our approch is extended MVC(Model-View-Controller) Framework and Remote-Object-Method-Invocation for hyperlink space in network. Also we show new typed infomation retrieval software-agent using dynamic OOPL(Object Oriented Programming Language) of Smalltalk.

1．はじめに

我々の周りには多種多様な文書情報が多数あるが，有効な形で関連付けて保存されていない等の理由により折角のデータが死蔵されているため，文書情報を体系的に整理し誰でもが有効活用出来るような情報の共有化を支援する情報管理システムが強く求められている．そこで，文書情報へのアクセスに注目し，文書ファイル資源をオブジェクト指向ファイルシステム化し，それに対するハイパーメディア操作が出来るGUIとを協調連係する「Ｗｒａｐｐｅｒ情報モデル」に基づくハイパーリンク型情報管理システムの開発を，動的オブジェクト指向言語Smaltalkにより進めてきた[1][2]．メディア情報とリンク情報との陽な分離，ハイパーリンクの一貫性管理という特徴を有する本方式をネットワーク環境下でも実現するためのWeb化及びCORBA化の２つの方式検討をした．これらの方式の本質が，ＭＶＣフレームワークの分散型への拡張，及び，オブジェクト指向に於けるメソッド呼び出しの遠隔化（分散オブジェクト化）であることを示す．また，開発言語の動的性を活かしたソフトウェアエージェントによる大域検索方式についても述べる．

2．ハイパーリンク型情報管理システム

2.1 Ｗｒａｐｐｅｒ情報モデル

動的オブジェクト指向言語Smalltalkのビジュアル開発環境であるVisualWorksを利用して開発を行った．

2.1.1 オブジェクト指向ファイルシステム化

Wrapper[3]は既存システムからのデータ変換・エミュレーションを意味するデザインパターンの一手法であるが，本モデルでは，既存のファイルシステム利用形態を包含しつつ，情報の付加価値化をより効果的に行なうための規格化された属性を持つオブジェクトでカプセル化するオブジェクト指向ファイルシステムとして利用をしている．

Wrapperの属性には，文書名，文書間リンク，キーワード等があり，文書ファイル本体とリンク情報の陽な分離方式を実現している．HTML文書のように，文書ファイル本体の中にパス名などの物理的な情報で静的リンク情報を直接に埋め込む方式は情報管理の観点からは大きな問題があるが[4][5]，文書ファイル本体とは別にリンク情報を管理することで，自由度の高いリンクを実現出来る．また，開発言語の動的性を活かすことで，データを保持したままオブジェクトの属性は段階的に拡張が可能である．

2.1.2 ハイパーリンクの一貫性管理

オブジェクト指向の概念をハイパーメディアモデル上に展開し，多種多様な文書をオブジェクトとして扱い，オブジェクト間にリンク(1:N)が付与される．内部的には双方向リンクが設定されるので，逆方向リンクの巡航やリンク元／先文書の削除，分類階層の移動に伴うリンク情報の一貫性管理が出来る．また，このリンクには属性・役割を付与出来るので，通常のハイパーメディアモデルに於けるリンクの概念より豊富な情報量を持つことで情報管理の能力を高めることが出来る．

2.2 メタクラスを用いた文書型の設定方式

文書に型を導入し，それに対応する組込エディタを開発した．例えば，キーワードもキーワード文書という文書型で表現している．

オブジェクト指向システム開発では，クラスライブラリ化・差分プログラミングが大きな特徴であるが，データ構造は無条件に継承されてしまうので，注意深い設計が必要となる（一方，メソッドはオーバーローディング等で対応出来る）．特に，マルチメディア情報を持つ不定形な文書に対しては予めデータ構造を決めることが出来ないため，データ構造を如何にフレキシビリティに保持するかが重要となる．そこで，Smalltalk システムが提供するクラス階層に基づくインスタンス変数の継承ではなく，メタクラスに自分達が与えたインスタンス変数の継承を使うことで，個々のクラス毎に同じ名前で異なる領域をクラス変数で持つことが出来るようになり，文書の型毎に最適なデータ構造を与えることが出来る．

3．Ｗｅｂ化対応システムの開発

3.1 ＭＶＣフレームワークの概要

ＭＶＣフレームワーク[6]，対話型グラフィックスシステムを構築する際の規範的モデルとしてSmalltalk-80で提案されたものであり，Model-View-Controller の３つ組のオブジェクトから構成される仮想型複合オブジェクトである．問題領域の構造を表現するアプリケーション固有部と，図形表示インタフェースに係わる処理部とを分離することの有効性を示た．

VisualWorksでは，よりプラガブルなＭＶＣとするために拡張がされており，データの格納と処理を行う情報モデルを，ドメインに関係するデータの定義と処理を行うドメインモデルと，UIオブジェクトとドメインモデル間の情報とサービスの提供をするアプリケーションモデルとに層別化される．つまり，この分離は，Modelオブジェクトが持つ，アプリケーションのデータ格納操作を行う部分と，アプリケーション特有の方法でデータを表示操作の２つの役割を反映している．

3.2 Ｗｅｂ化に際しての問題点

VisualWorksで開発した既存のツールをWeb化でも利用するためには，先ず，VisualWorksのウィンドウ管理機構であるＭＶＣフレームワークをイントラネットで実現する方式が必要となる．つまり，オブジェクトメモリ上でポインタにより束縛されているオブジェクト群を，コネクションレス型の通信を行うHTTPの制約のもとで，Model（Webサーバ上のオブジェクト）とView（Webブラウザ上のHTML文書）の間で，照合を取る方式が必要となる．

また，Webブラウザから，VisualWorksで開発した既存のツールの機能を利用すると共に，類似したGUI操作を実現する方式が必要となる．

更に，Webブラウザからデータ更新利用を行うためには，Webサーバアプリケーションで生成・管理されているオブジェクトと，Webブラウザ上に表示されているHTML文書との対応関係を照合する方式と，複数の利用者による競合を防ぐ排他制御が必要になる．

3.3 オブジェクト指向開発実行環境VisualWave

VisualWaveはVisualWorksにWebサーバとの連係機能を提供するためのクラスを拡張したものであり，Visualworksで開発されたアプリケーションのWeb化を支援するものである．つまり，VisualWorksで開発済みのアプリケーションをWebサーバアプリケーションとして利用できる．

VisualWaveではHTML文書やCGIプログラムのジェネレータ機能を有している．更に，Smalltalkという高い記述能力を持つオブジェクト指向言語により，ソフトウェア部品の再利用化と共に，サーバー側の複雑なWebアプリケーションを開発することが出来る．

Webブラウザからの入力や検索などの要求は，CGIのプロトコルに従いWebサーバが受け付けた後に，VisualWaveが備えるCGI-Relay方式で転送され，VisualWaveはデータベース等の操作を行い処理結果のデータをHTML文書に動的に変換をして，Webブラウザに返送する．

3.4 分散型ＭＶＣフレームワーク

ＭＶＣフレームワークをネットワークに拡大した場合の拡張系についての検討がされてきているが[7]，コネクション型のネットワーク環境下を想定した場合である．

HTTPプロトコル自体はコネクレションレス型の通信を行い，Webブラウザ上に表示されているHTML文書はオブジェクトではないので，どのアプリケーションの，どの時点の状態のオブジェクトであるかが特定出来ない．VisualWaveでは，これを識別するキーとしてセッションIDとページIDを発番し，HTML文書に変換する際には，ページを構成するウィジェットを表すフォームなどと共に，これらのIDをhidden属性を持つフォームとして書き込むことが出来る．動的に生成し転送したHTML文書と，Webブラウザから送信されてきたHTML文書のキー（セッションIDとページID）を照合し，一

致した場合には，そのHTML文書のページは，同じドメインモデルに対応するViewであるとみなすことで，分散型のＭＶＣフレームワークを構築することが出来る．

3.5 差分プログラミングによるツールのWeb化

プログラミングにおいて殆どの部分は同じで，一部だけ異なったモジュールを作成したいことがよくある．元のモジュールのプログラムをコピーすることなしに，新しく作りたいモジュールを新しい名前で定義し，そこには元のプログラムと同じであることを宣言するだけで上述のことが出来れば大変便利であり，このような方法を差分プログラミングという[6]．

VisualWorksで開発した従来システムのソースファイルをVisualWave開発ツールでコンパイルすることにより，開発済みプログラムの全ての機能をVisualWaveで使用できるようになる．しかし，VisualWorksはコンソールでの使用を前提としているためにWebブラウザから使うことは出来ない．このため，個々のツールに関して，それを構成するアプリケーションモデルクラスのサブクラス化（差分プログラミング）を行ないWeb化できるように特殊化した．差分プログラミングによるツールのＷｅｂ化のシステム概念図を図１に，画面の比較例を図２に示す．

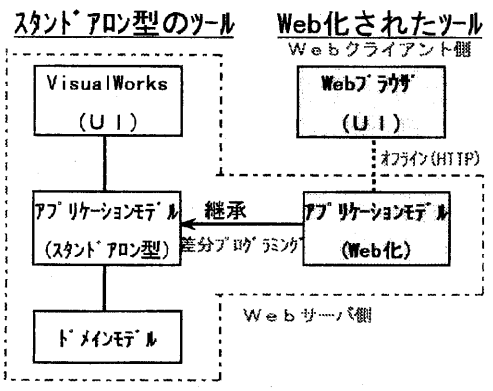

図１．差分プログラミングによるツールのWeb化

3.6 メソッド呼び出しの遠隔化

HTTPではイベントは常にWebブラウザから起き，Webサーバはそれに対して返送する仕組みであるので，VisualWorksのGUIに相当するトリガとなるアクションボタンをWebブラウザ上のページに加えることで対応する．Webブラウザ上のアクションボタンを送信することで，間接的に，これに対応するサーバ側のオブジェクトのメソッドを呼び出すこと（分散オブジェクト化）が出来る．つまり，VisualWorksで開発済みのアプリケーション（Webサーバアプリケーションとして利用）をWebブラウザからも共通利用できる．

3.7 HTML文書の動的性の一元管理

データベースの検索結果を反映したHTML文書を作成するために，HTML文書の動的性が強く望まれている．インタラクティブなWebページをより容易に作成するためにMicrosoft社とNetscape社が提案したダイナミックHTMLはWebブラウザ上で動作するものであるが，現状，プラットフォームの限定，ベンダー間での互換性に欠けるなどの問題点がある．本方式では，サーバ側の統一管理のもとでHTML文書を動的に作成し転送する方式であり，方式の比較を図３に示す．

3.8 オブジェクト共有による排他制御

Web化では，組み込みエディタをWebブラウザ経由でも使えるように拡張しているため，複数の組み込みエディタから同時に同一の格納文書が開かれる可能性があり，ここで競合が発生する．

本システムでは，基本的なセッション機構(セションID＋ページID)をベースに，個々のエディタについて，リード・ライト・モードとリード・オンリ・モードを新たに設けて，リード・ライト・モードにあるエディタのオブジェクトが常に1つ以下であることを保証し，更に，HTTPとVisualWorksで異なるウィンドウの開閉（アプリケーションの開始／終了）方法の調整，個々のセッションにタイマを設置するなどをして，データ更新利用で必要となる排他制御を実現している．

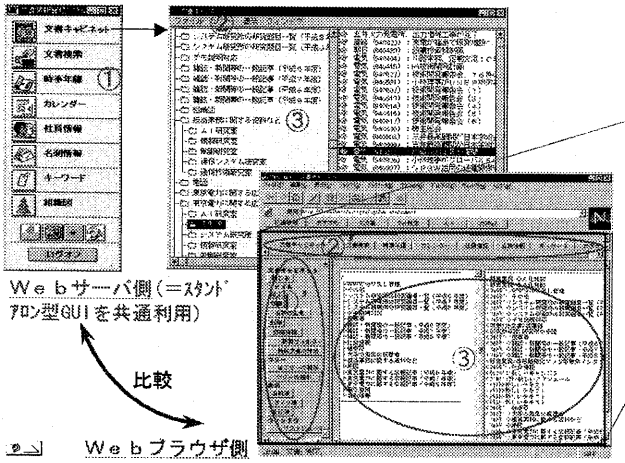

図２．画面の比較例（スタンドアロン型とＷｅｂ化）

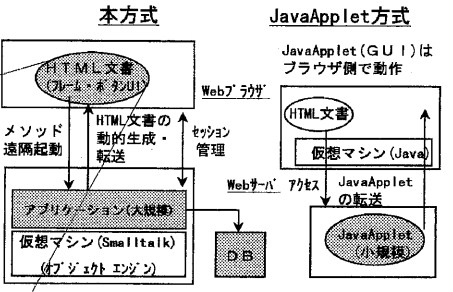

図３．HTML文書の動的性の比較図

４．ＣＯＲＢＡ対応化システムの開発

4.1 ＣＯＲＢＡ化に際しての問題点

スタンドアロン型の構成方式のWrapper情報モデルを複数のマシン間でも実現をするために，分散オブジェクトを用いて再設計し実装する必要がある．また，ネットワーク環境下で検索を行う際の拡張性・柔軟性を持つ新たな方式を検討する必要がある．

4.2 分散Smalltalk[8]

分散Smalltalkは，オブジェクト指向分散アプリケーション開発実行環境であり，VisualWorksにCORBA(分散オブジェクト間通信路の標準規格，Common Obejct Request Broker Architecture)の2.0仕様に準拠した分散機能を提供するためのクラスを拡張したものであり，VisualWorksで開発されたアプリケーションのCORBA化を支援するものである．分散Smalltalkの設計指針は単純明快であり，ネットワークで繋がるマシンの上に存在するという拡張を行ったものである．

4.3 分散型ＭＶＣフレームワーク[9]

分散Smalltalkでは，VisualWorksのＭＶＣフレームワークを構成する２つのモデル（ドメインモデルとアプリケーションモデル）が分散オブジェクトとして別々のコンピュータ上にある場合もサポートするための拡張ＭＶＣフレームワークを既に備えている．具体的には，ドメインモデルに対応するＳＯ(Semantic Object)，アプリケーションモデルに対応するＰＯ(Presenter Object)を分散オブジェクト環境下で分離した構造を採用している．図４に，分散型ＭＶＣの概観を示す．

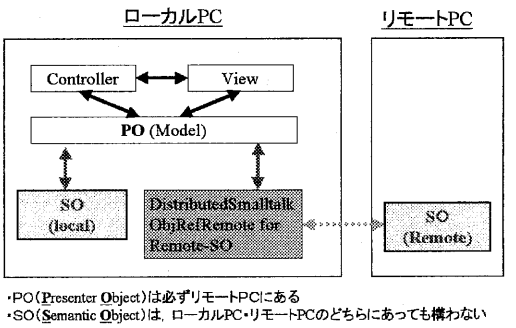

図４．分散型ＭＶＣの概観(分散Smalltalk)

分散Smalltalkの基本的なクラスは以下の通りである.

4.3.1 ORBObject

分散Smalltalk用のORBであり，分散サービスを提供するクラスである．クライアントが発行するリクエストを目的のオブジェクトに届け，その応答をクライアントに返却する．アプリケーション開発で，ネットワーク関連が見える所に相当する．ORBObjectの向こう側はブラックボックスのままでもプログラミングできるが，エラー解析時には中まで見る必要がある.

4.3.2 分散SmalltalkRepository

分散Smalltalk用のインタフェース定義を提供するクラスである．IDLの定義は，分散Smalltalk Repositoryのメソッドに記述する．IDLのコンパイラはメソッドのアクセプトと同時に行われる.

4.3.3 分散SmalltalkSemantic（SO）

ドメインモデルのサブクラスでDB上のオブジェクトに相当する．オブジェクトはローカル，リモートの何処にあっても構わない．SOオブジェクトの表示は，対となるPOオブジェクトで行う.

4.3.4 分散SmalltalkPresenter（PO）

アプリケーションモデルのサブクラスで，SOオブジェクトを表示するクラスである．ローカル上にオブジェクトが生成され，SOオブジェクトとコミュニケーションを取りながら表示される.

4.3.5 分散SmalltalkObjRef

分散環境でのORBのRPCメカニズムを提供するクラスである．分散SmalltalkObjRefオブジェクトはローカルPCにあり，メッセージを受信すると，参照しているリモートPCのオブジェクトへメッセージを送信する仕組みをもつ.

4.4 Wrapperオブジェクトの分散オブジェクト化

分散Smalltalkにはオブジェクト同士をリンクさせるための機構を４種類(ContainmentLink, ReferenceLink, DesignationLink, WeakLink)予め持っている.

Wrapper情報モデルでは，文書間には双方向リンクが設定され，リンクの一貫性管理が行われるが，ContainmentLink機構が双方向モデルであり，オブジェクトの動的な移動や削除にも対応していることに注目し，Wrapper情報モデルをこのサブクラスとして設計し実装をした．これにより，複数のマシン間でのリンク付け，リンクの一貫性管理は実現できる.

2.1.3で上述したように，本システムでは，文書に型を導入し，情報管理単位を全て文書として表現をしている．例えば，キーワードをキーワード文書，キーワードを包含するカテゴリをカテゴリ文書として定義をし，必要なデータ構造，メソッド，制約を持つ．文書とキーワードとの参照関係は，内部的には，文書同士の参照関係と同じである．また，キーワード文書，カテゴリ文書は同一マシン上にある必要はない.

分散オブジェクト間のリンク例と，その参照関連図を，図５，６に示す.

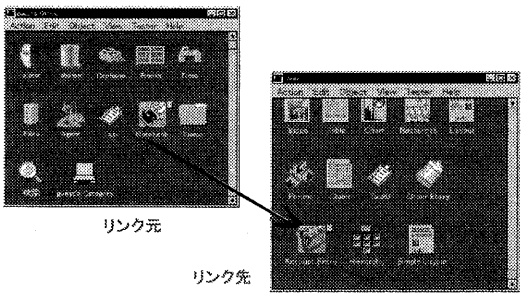

図５．分散オブジェクト間のリンク例

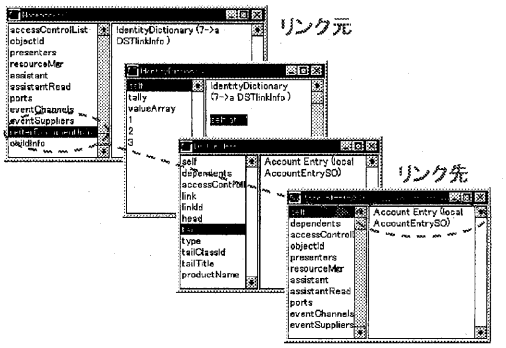

図６．リンクの参照関連図(ｲﾝｽﾍﾟｸﾀﾂｰﾙ利用)

4.5 分散Smalltalkの問題点

4.5.1 分散オブジェクトの永続化

分散オブジェクトの永続化(ディスクへの保存)は，Smalltalkのイメージで行っている．そのため，複数のＰＣ間での分散オブジェクトの整合性を図るためには，複数のイメージのスナップショットを取るタイミングを揃える必要がある。また，イメージが起動されていなければそのイメージにあるセマンティックオブジェクトは利用できないので，全てのクライアントとサーバのイメージが起動されている必要がある。

4.5.2 ＩＤＬの変数の型

分散Smalltalkでは，Smalltalkのメソッドから IDLを自動生成する機構(IDL Generator)があるが，IDL Generatorでは全ての変数がSmalltalkObject型(any型)になってしまうため，CORBAの目的を考えると，データの型は明確にすべきである．

4.5.3 分散型ＭＶＣの実装の不完全さ

分散Smalltalkでは様々なマルチメディアオブジェクトをサポートしているが，依存性を利用していないオブジェクトがあり，Modelへの新たなViewの登録，ModelからViewへの表示の要求方法が統一化されておらず，ハイパーメディア型操作インタフェースを統一的に開発するのが難しい．

5．ソフトウェアエージェントを用いた検索方式

システムの拡張性・柔軟性を高めるために，プログラミング言語の動的性が注目されている[10][11]．Smalltalkでは，ブロッククロージャと呼ばれる，関数閉包と遅延評価が可能なオブジェクトを定義できる．Smalltalkの多彩なプログラム制御構造を作り出す鍵であり，whileやif～then～elseなどの決まり切った制御構文が文法には存在しないにも係わらず，予約語に相当するかのように利用できるのは，ブロッククロージャのお陰である．これを情報検索に利用したソフトウェアエージェント[12][13][14]を試作をした．

2.1.3で上述したように，本システムでは，文書に型を導入し，情報管理単位を全て文書として表現をしており，検索では，検索文書，タスク文書（エージェントが仕事を行うための指示文書，中身はブロッククロージャ）の２種類を定義した．

利用者側の操作は以下の通りである．

- 検索対象としたいリモートマシンを，ローカルの検索文書に束縛する(drag&drop操作)
- 検索条件パネルの起動（文書型，文書名，ｷｰﾜｰﾄﾞ等の検索条件の指定）
- 検索結果の表示確認

この裏で，以下の処理が間接代理実行される．

- 検索条件からタスク文書を自動生成
- タスク文書を検索エージェント(ローカル)に投入
- 検索エージェント(ローカル)は，タスク文書の内容に従い，順次，検索エージェント(リモート)を起動し，タスク文書を送信
- 検索エージェント(ローカル)は，検索エージェント(リモート)の処理結果を集約して，検索文書に格納

検索条件の指定，タスク文書の自動作成，エージェントの処理依頼（タスク文書の投入），検索結果の表示一覧等は，検索文書が持つべき機能としてまとめることが望ましく，文書にオブジェクト型を導入することの利点がある．

ローカル側のマシンに負荷をかけないように，エージェント間の通信方法は非同期処理とし，リモートのエージェントを遠隔呼び出して処理をさせるようにしている．

分散Smalltalkには，このブロック式を使ったインタフェースエージェントの構築や操作を支援するSmalltalk Agentが提供されているが，現状，Microsoft Agentと同様に単なるフロントエンド(見えるエージェント)でしかなく[15]，実際の問題解決処理部分は書かなくてはならない．本方式では，利用者からの検索条件から問題解決処理部分をブロッククロージャとして自動作成し，応用システム間同士のエージェントの協調処理[16]，検索結果の集約表示まで，間接代理実行を行うところまで機能を高めている．

検索エージェントの処理概観を図７に示す．

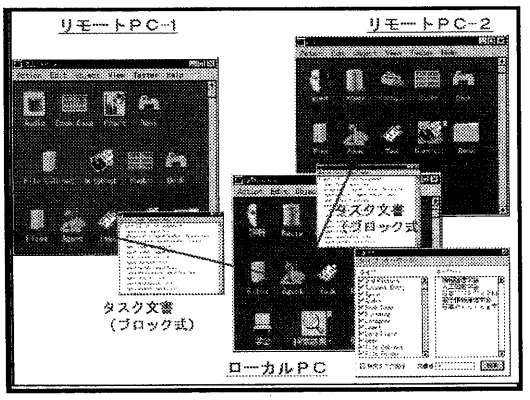

図７．検索エージェントの処理実行の概観

６．おわりに

メディア情報とリンク情報との陽な分離，ハイパーリンクの一貫性管理という特徴を有する本方式のWeb化及びCORBA化の２つの方式開発をした．これらの方式の本質が，MVCフレームワークの分散型への拡張，及び，オブジェクト指向に於けるメソッド呼び出しの遠隔化(分散オブジェクト化)であることを示した．また，開発言語の動的性を活かしたソフトウェアエージェントによる大域検索方式を開発した．

今後の計画として，以下を考えている．

- ネーミングコンテキストを利用した検索性能の向上(CORBA化)
- Smalltalkの拡張クラスライブラリである３つのインタフェース（Web，CORBA，DCOM）を利用し，Smalltalkオブジェクトを中心とした分散オブジェクトのサービス利用の融合化[17]
- ハイパーメディアシステムに於けるインタフェースエージェントの構築や操作を支援するツール化[18]

参考文献

[1]窪野：差分ﾌﾟﾛｸﾞﾗﾐﾝｸﾞを用いたﾊｲﾊﾟｰﾒﾃﾞｨｱ型文書情報管理ｼｽﾃﾑのｲﾝﾄﾗﾈｯﾄ化，情処学会ｵﾌﾞｼﾞｪｸﾄ指向ｼﾝﾎﾟｼﾞｳﾑ，Sept. 1997

[2]窪野：C/Sとｲﾝﾄﾗﾈｯﾄとの融合を指向したﾊｲﾊﾟｰﾒﾃﾞｨｱ型文書情報管理ｼｽﾃﾑ，情処学会DBS111-7, 1997

[3]Erich Gamma, et al(本位田、吉田和樹監訳)：ｵﾌﾞｼﾞｪｸﾄ指向における再利用のためのﾃﾞｻﾞｲﾝﾊﾟﾀｰﾝ，ｿﾌﾄﾊﾞﾝｸ，1995

[4]Spcial issue on Hypermedia,Communications of the ACM,Vol.37,No.2,1994

[5]田中：ﾏﾙﾁﾒﾃﾞｨｱとﾈｯﾄﾜｰｸがもたらす新しい情報共有基盤，情報処理学会ADBS'96論文集，1996

[6]所，松岡，垂水：ｵﾌﾞｼﾞｪｸﾄ指向ｺﾝﾋﾟｭｰﾃｨﾝｸﾞ，岩波，1993

[7]村田，増市，荒谷：ﾕｰｻﾞｲﾝﾀﾌｪｰｽ構築のためのﾏﾙﾁｴｰｼﾞｪﾝﾄﾓﾃﾞﾙ，日本ｿﾌﾄｳｪｱ科学会，MACC'91

[8]Timothy W. Ryan(植野・多田訳)：分散ｵﾌﾞｼﾞｪｸﾄﾃｸﾉﾛｼﾞｰ，ﾌﾟﾚﾝﾃｨｽﾎｰﾙ，1997

[9]W.Lalonde,J.Pugh : Building an appication using HP Distributed Smalltalk,JOOP,Vol.6,No.6,1994

[10]苫米地：進化ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語Common Lisp，bit，Vol. 30, No. 3-4, 1998

[11]D.Ingals,T.Kaehler,J.Maloney,S.Wallace,Alan Kay: "Back to the Future:The Story of Squeak,A Practical Smalltalk Written in Itself",ACM OOPSLA'97

[12]石田：ｴｰｼﾞｪﾝﾄを考える，人工知能学会誌，Vol. 10, No. 5, 1995

[13]Spcial issue on Intelligent Agent,Communications of the ACM,Vol.37,No.7,1994

[14]西田：「情報共有のためのﾈｯﾄﾜｰｸ・ｴｰｼﾞｪﾝﾄ」ﾁｭｰﾄﾘｱﾙ資料，日本ｿﾌﾄｳｪｱ科学会，Nov. 1997

[15]田村，若月：ｻｲﾊﾞｰｴｰｼﾞｪﾝﾄ(1)，OplusE，No. 217, Dec. 1997

[16]大沢，横尾：「ﾏﾙﾁｴｰｼﾞｪﾝﾄ」ﾁｭｰﾄﾘｱﾙ資料，日本ｿﾌﾄｳｪｱ科学会，Mar. 1998

[17]Applied Reasoning Systems Corporation: An Applied Reasoning White Paper Route1: The Fastest Way to the Web!,Sep.1996

[18]J.Alfedo,et al:HyperActive,ACM Trans. on Comp.Interact,1994