グループウェア 37－9
(2000. 10. 19)

階層的キーワードに基づく名前管理手法とそれに基づくファイル共有手法

轟木 伸俊† 多田 知正† 樋口 昌宏‡ 谷口 健一†

†大阪大学
大学院基礎工学研究科 情報数理系専攻
〒560-8531 大阪府 豊中市 待兼山町 1-3

‡近畿大学
理工学部 電気工学科
〒577-8502 大阪府 東大阪市 小若江 3-4-1

あらまし 従来ファイルシステムにおいては，階層的名前管理が一般に用いられている．これに対し本研究では，階層的キーワードベースの名前管理を考えている．これは，ファイルに階層的に管理されたキーワードの集合を付与することによりファイルの識別を行うというものである．本稿では，この階層的キーワードベースの名前管理におけるファイルの共有について考える．複数のユーザがファイルを共有する場合，全員が一つの名前空間を共有すると，ファイル分類の自由度が制限され，不便であるため，各ユーザが自分自身の名前空間を構成し，それらを統合するという方法が望ましいと考えられる．階層的キーワードベースの名前管理は従来の階層的名前管理と比べて，この様な状況により適していることを示す．また我々の作成したファイル名管理システムのプロトタイプにおける，名前空間の統合の実装について述べる．

和文キーワード ファイルシステム，キーワード，分類，名前管理手法，データベース

Hierarchical-keyword-based Naming Scheme and File Sharing based on it

Nobutoshi Todoroki† Harumasa Tada† Masahiro Higuchi‡ Kenichi Taniguchi†

†Graduate School of Engineering Science
Osaka University
Toyonaka, Osaka 560-8531 Japan

‡School of Science and Engineering
Kinki University
Higashi-Osaka, Osaka 577-8502 Japan

Abstract In file systems, hierarchical naming is generally used. We are studying another naming scheme, called hierarchical-keyword-based naming. In hierarchical-keyword-based naming, a file is named by a set of keywords and keywords are organized hierarchically. In this paper, we discuss file sharing in hierarchical-keyword-based naming. In the case that multiple users share files, it is inconvenient to share a global name space because file organization by users is somewhat restricted in some degree. It is more desirable for users to have their own name spaces, and integrate them for file sharing. We show hierarchical-keyword-based naming is more suitable for this purpose than usual hierarchical naming. We also describe how to implement integration of name spaces in our prototype of filename management system.

英文 **key words** Filesystem, Keyword, File Organization, Naming Scheme, Database

1. まえがき

二次記憶装置の容量の増大に伴い，ファイルシステムの管理するファイルの数も膨大なものとなっており，目的のファイルを探し出すことがより難しくなりつつある．また，最近は画像ファイルや音声ファイルなどのマルチメディアファイルが一般化してきている．これらのファイルはテキストファイルと違い，全文検索を用いたファイル検索手法を適用することができない．したがって，ファイルの名前によるファイルの管理がより重要になってきている．

従来のファイルシステムでは，ファイルは木構造のディレクトリによって管理されており，ユーザはパス名によってファイルを指定する．このような名前管理を階層的名前管理という．

階層的名前管理は，実装が容易であるなどの利点がある．しかしファイル数の増加に伴って，ディレクトリ構造が深く複雑になると，目的のファイルを探し出すことが困難になる．このために，大量のデータを扱うことの多いデータベースシステムでは一般に階層的名前管理は用いられない．多くのデータベースシステムでは，属性を付与することでデータを管理する．ユーザはクエリを用いてデータを取得する．このようなタイプの名前管理は，属性ベースの名前管理と呼ばれる．

より適切なファイル管理のために，属性ベースの名前管理を階層的の名前管理に導入することが試みられている [2, 3, 4]．

我々はこのような名前管理の一つとして階層的キーワードベースのファイル名管理 (以下，階層的キーワード名前管理) を考えている．これは階層的名前管理と属性ベースの名前管理の利点を合わせ持つ名前管理である．階層的キーワード名前管理では，ファイルにキーワードを付与することで分類を行う．ファイル指定の際には，キーワードを任意の順序で並べることができ，またそれは一部のキーワードでもかまわない．キーワードは階層的に管理されているため，キーワード集合はディレクトリ木のような階層構造をとっている．ユーザは目的のファイルに付与されたキーワードがわからないとき，キーワード階層を探索することでキーワードを見つけ出すことができる．

階層的キーワードベースの名前管理におけるファイルの共有について考える．複数のユーザがファイルを共有する場合，全員が一つの名前空間を共有すると，ファイル分類の自由度が制限され，不便である．各ユーザが自分自身の名前空間を構成し，必要に応じて，それらを統合するという方法を用いると，ユーザの名前空間を自然な形で統合することが可能であり，望ましいと考えられる．階層的キーワード名前管理では，ファイルの共有により適していると考えられる．

本稿では，提案する階層的キーワード名前管理のもとで，複数のユーザによるファイルの共有について述べる．ファイルの共有における，階層的キーワード名前管理の利点と問題点をあげ，問題を解決するための手法を検討した．

本稿では，2 節で従来の名前管理について述べる．3 節では，提案する階層的キーワード名前管理について述べる．4 節では，提案する名前管理がファイルの共有に適していることを示す．5 節では，階層的キーワード名前管理に名前空間の統合を導入することに際しての問題点を挙げ，その問題点の解決手法を与える．6 節では，本研究で実装しているプロトタイプシステムにおけるファイルの共有を実現する手法について述べる．最後に，7 節で結論を述べる．

2. 従来の名前管理

ここでは，従来の名前管理手法である階層的名前管理と，属性ベースの名前管理について述べる．

2.1 名前管理手法と名前空間

ファイルの名前とは，ファイルを指定するために用いられる文字列のことをいう．従来の UNIX ファイルシステムにおける，絶対パス名，相対パス名は共にファイルの名前である．ファイル名とは，ファイルを識別するためにファイルに付与される文字列である．UNIX では，それを格納するディレクトリ内で一意の文字列 (`main.tex` や `fig.jpg` など) をいう．

ファイルの名前の集合はなんらかの構造を持ち，名前空間と呼ばれる [1]．ファイルの名前の表記法とその解釈のしかたを規定するものを名前管理手法という．ファイルの名前をファイル名と区別するために，以降ではファイルの名前をファイル指定と呼ぶ．

2.2 階層的名前管理

木構造で管理されるディレクトリに登録することでファイルを管理する従来の名前管理は階層的名前管理と呼ばれる．

それぞれのファイル (ディレクトリを含む) は，それを格納しているディレクトリ内で一意のファイル名をもつ．ファイルは一連のディレクトリ名とファイル名の並びであるパス名によって指定される．パス名はそのファイルが格納されている位置を示している．このような名前管理によるファイル名の集合は階層的名前空間と呼ばれる．

階層的名前管理の利点は，以下のようなものである．

- ファイル探索の際に構造を把握しやすい
- システムが名前からファイルを得る際に探索範囲が小さくてすむ
- カレントディレクトリの概念により相対的なファイル指定が可能
- ファイル探索の際ディレクトリ構造が一種の道案内の役割を果たす

2.3 属性ベースの名前管理

データベースでは属性を付与することでデータを管理している．ユーザは $(owner = todoroki) \wedge (type = text)$ のようなクエリによりデータを指定する．クエリはデータの集合を表す名前と考えることができ，その集合は階層構造を持たない名前空間を構成する．このような名前管理手法を属性ベースの名前管理という．

属性ベースの名前管理の利点は以下のようなものである．

- ファイル指定の際に属性の順序を気にしなくてよい
- 一部の属性でファイル指定ができる
  この特徴は次のような利点を与える.
  - 多くの属性を付与することで詳細な分類ができる
  - 付与された属性の一部でも記憶していれば素早いファイル検索ができる
  - 属性を追加しても以前と同じファイル指定ができる

3. 階層的キーワード名前管理

3.1 概要

階層的キーワード名前管理は，階層的名前管理と属性ベースの名前管理の双方の特徴を持つ名前管理である.

階層的キーワード名前管理では，それぞれのファイルにはファイル名の他に，一つ以上のキーワードが付与される．キーワード集合は階層構造を構成する．このようなキーワードを階層的キーワードと呼ぶ．階層的キーワードは階層的名前空間におけるパス名と同様に記述することができる．例えば */photo/animal/dog* のように記述する．ファイル指定は，*photo, dog, child//fig1.jpg* のように，コンマで区切られたキーワードの非順序リストとファイル名からなる．ファイル名はキーワードの非順序リストの最後に//で区切って書かれる．ファイル名を含まないキーワードのリストはファイルの集合を指定する．あるファイルに付与された全ての階層的キーワードのリストとファイル名からなるファイル指定を一意名と呼ぶ．一意名は階層的名前管理における絶対パス名に相当し，システム中でファイルを一意に指定する．キーワードのリストは非順序なので，一つのファイルの一意名は複数存在する．例えば，*/photo/animal/dog*, */photo/people/child//fig1.jpg* と，*/photo/people/child*, */photo/animal/dog//fig1.jpg* は同じファイルを表す一意名である.

3.2 ファイル指定

ファイル指定により指定されたファイル集合は，キーワードリストがマッチするファイルを全て含む．ファイル指定は複数のファイルを指定することがある.

「キーワードリストがファイルにマッチする」とは，キーワードリスト内のそれぞれのキーワードが，ファイルの階層的キーワードの少なくとも一つにマッチすることをいう．キーワードは，それが含まれる階層的キーワードにマッチする．例えば，キーワード **sports** は */vehicle/car/sports*, */sports/baseball*, */article/sports/swimming* 等にマッチする.

ファイル指定には階層的キーワードの一部を含むことができる．例えば，**/image/photo,skate/woman//photo-2.jpg** という記述も許される．この場合，*/image/ photo* と *skate/woman* にマッチする，*photo2.jpg* というファイルを指定する．*skate/woman* にマッチするキーワードの例を表 1に示す.

表 1 *skate/woman* にマッチするキーワード

マッチする	マッチしない
/sports/skate/woman	/sports/woman/skate
/skate/woman/photo	/skate/figure/woman

3.3 カレントファイル指定

階層的名前管理においてはカレントディレクトリの概念があり，ユーザは相対パスを用いてファイルを指定することができる．このような相対的なファイル指定を可能とするため，カレントディレクトリに相当するものとして，カレントファイル指定を導入する．これは現在ユーザが注目しているファイル集合 (以下，カレントファイル集合) を指定するキーワードのリストであり，通常はファイル指定に自動的に付加される．このようなファイル指定を相対ファイル指定と呼ぶ．カレントファイル指定を用いることで，ファイル名のみによるファイル指定が可能となる．カレントファイル指定を付加されないようにするには，ファイル指定の先頭に@をつける．このようなファイル指定を絶対ファイル指定と呼ぶ.

3.4 ファイルの分類

階層的キーワード名前管理においては，ファイルにキーワードを付与することで分類を行う．キーワードをファイルに付与するためには，そのキーワードがキーワード階層に存在しなくてはならない．すなわち，キーワードを付与する前にキーワード階層のどこかにそのキーワードを作成しておく必要がある.

階層的名前管理では，ディレクトリにファイルを移動することでファイルの分類を行なう．更に詳細に分類する際には，新たなサブディレクトリを作り，そこにファイルを移動する．詳細に分類するほどディレクトリ階層がより深く，複雑になる．ディレクトリ階層が深くなることは，ファイルの検索の際に問題となる．階層的キーワード名前管理では，ファイルに階層的キーワードを追加していくことで詳細な分類が可能である．したがって，キーワード階層をそれ程深くする必要はない.

例えば,「2000年の8月に撮った鶴の写真」を格納することを考える．階層的名前空間では，詳細に分類するためには，*/image/photo/2000/August/bird/crane* のようなディレクトリを作成し，ファイルを格納する．これに対し，階層的キーワード名前管理では，{*/image/photo*, */2000/August*, */bird/crane*} といったキーワードを付与することで分類が可能である.

3.5 ファイル検索

階層的キーワード名前管理では，ユーザが目的のファイルの一意名を知らないとき，カレントファイル指定を変更しながらファイルを検索する．最初に，既知のキーワードをカレントファイル指定に加え，カレントファイル集合のリストを閲覧する．もしカレントファイル集合が大きくリストから選び出すことが難しいならば，別のキーワードをカレントファイル指定に加えることでカレ

ントファイル集合を小さくする．カレントファイル集合に目的のファイルが含まれていないならば，カレントファイル指定からキーワードを削除し，新たな別のキーワードを加える．このとき，ユーザがどのキーワードを加えればよいのかを知らなければ，適切なキーワードが見つかるまでキーワード階層に沿って探索することになる．

階層的名前管理ではユーザが目的のファイルの正確なパス名を知らない時，そのファイルの格納されたディレクトリが見つかるまで探索を続ける．階層的名前管理では，階層が深くなりやすい傾向があり，目的のディレクトリを見つけるために多くの時間を要する場合がある．階層的キーワード名前管理では，それほど階層が深くならないため，目的のキーワードを見つけるのが容易である．

階層的名前管理では親ディレクトリが子ディレクトリの情報を隠蔽してしまうため，たとえパス名の一部が既知であっても，それを探索に利用することは難しい．階層的キーワード名前管理では，詳細な分類のために多くのキーワードを付与することができ，その中の一部のキーワードでも記憶していれば，目的のファイルが含まれたファイル集合を得ることができる．

このため，階層的キーワード名前管理では，正確なパス名を記憶しておく必要のある階層的名前管理と比べて，ユーザの負担を軽減することができる．

4. ファイルの共有

階層的キーワードベースの名前管理におけるファイルの共有について考える．

複数のユーザが存在する環境において，一般にユーザはそれぞれ自分の名前空間を持ち，その中で恣意的にファイル名を決定する．複数ユーザでファイルを共有する際には，どのような形で名前空間の共有を行うかが問題となる．名前空間を共有する方法は2通り考えられる．一つは全てのユーザのファイルを一つの構造で管理するというものである．この場合，それぞれのユーザが自由に名前をつけると，全体の構造が分かりにくいものとなる．これを避けるために，名前をつける際にユーザが従うべき何らかの規則を設けると，ファイル分類の自由度が制限されることになり，不便になる．もう一方の方法は，それぞれのユーザごとに管理するというものである．この方法を用いると，階層的キーワード名前管理においては，ユーザの名前空間を自然な形で統合することが可能である．したがって，後者のようにユーザ毎にファイルを統合する方がよいと考えられる．

階層的名前管理では，ファイルを利用する際には，そのファイルのパス名が必要になる．パス名はディレクトリ名のリストを含むファイル名で表されるが，それらの間には順序があり，ファイルのアクセスには正確な記述が必要になる．しかしながら，各ユーザのディレクトリ階層は恣意的に構成されており，その構造を把握しておくことは困難である．

階層的キーワード名前管理では，目的のファイルに付与された適切なキーワードを記憶していれば，検索が容易になる．ユーザ自身で付与したキーワードであれば，それを記憶している可能性が高い．しかし，他ユーザのファイルの場合，適切なキーワードが既知であるとは限らない．この時は，階層的名前管理の場合と同様に，対象となるユーザのキーワード階層を探索する必要がある．しかしながら，目的のファイルを発見した際に，そのファイルの名前の適切のキーワードを記憶しておくことで，以降はそのキーワードで目的のファイル集合を指定することができる．階層的名前管理では，階層構造を正確に覚えておかなければ，再びディレクトリ構造の探索が必要となる階層的キーワード名前管理はファイルの共有により適しているといえる．

5. 名前空間の統合における問題点

階層的キーワード名前管理において名前空間の統合を行う際に，以下のような問題が考えられる．

1. 階層的キーワード名前管理では，名前空間を統合すればするほど探索空間が拡大する．探索空間が広範囲になることでシステムにかかる負担が大きくなり，性能低下につながる．

2. ユーザは恣意的にキーワードを作成する．複数のユーザが存在する環境では，同じ意味のものでも，ユーザ毎に異なるキーワードが作成される場合がある．例えば，写真を示すキーワードとして，**photograph** や **photo**，あるいは，**shasin** といったキーワードが作成されることがある．これらは同じものを指すためのキーワードであるが，綴りが異なるため，キーワードとしては別個に扱われてしまう．

3. ファイルにキーワードが少数しか付与されていない場合，他のユーザが利用する際には不都合な場合がある．

上記の問題に対して以下に述べるようなしくみを提供する．

1. 名前空間の明示的な統合
   ユーザが必要とする時に明示的に他のユーザの名前空間の統合を行い，またユーザが必要としなくなった時に明示的に切り離すしくみを提供する．

2. キーワードの読み替え
   名前空間を統合したことで，同じ意味を示す複数の異なったキーワードが存在する場合，それらをある一つのキーワードに読み替えるようにする．どのキーワードをどのように読み替えるかは，ユーザに依存する．したがって，ユーザに対し読み替えの対象となるキーワードを登録するしくみを提供する．読み替えの対象となるユーザの名前空間を統合する際に，登録された内容に従い読み替えを行う．

3. 他ユーザのファイルへのキーワード付与
   ユーザのキーワード付与が適切でない場合の対策として，他ユーザのファイルへのキーワード付与を許可する．しかし他ユーザの名前空間を変更することは許されない．他ユーザの名前空間に影響を与えることなく，キーワードの付与を行うしくみを提供する．

他ユーザが付与したキーワードも通常のキーワードと平等に扱う．例えばユーザAが，あるファイルに付与したキーワードを全て削除した時，ユーザAは，もはやそのファイルにアクセスすることができなくなる．しかし，そのファイルに対して他のユーザBがキーワードを付与していた場合は，ユーザBはそのキーワードを用いてアクセスすることが可能である．

6. プロトタイプへの適用

6.1 UNIXファイルシステム上のプロトタイプ

本研究では，提案する名前管理をUNIXファイルシステム上で実現するプロトタイプシステムを実装している．

6.1.1 構成

プロトタイプシステムは，既存のUNIXオペレーティングシステム上に実装する形をとっており，通常のシェル上で実行される．ユーザはコマンドと，その引数としてファイルを指定する．システムはファイル指定を受け取り，対応するファイル集合の絶対パスに変換し，コマンドと共にシェルに渡すという作業を行う．

6.1.2 キーワードの管理

プロトタイプシステムでは，キーワードの階層は，ディレクトリの階層として実装される．したがって，それぞれの階層的キーワードは実際のディレクトリに対応している．階層的キーワードには，キーワードIDと呼ばれる一意な識別子が割り当てられている．

キーワードはテーブルによって管理される．これは，キーワードIDよりキーワードを得るためのものである．このテーブルをキーワードテーブルと呼ぶ．その例を表2に示す．

表 2 キーワードテーブル

ID	階層的キーワード
1	/image
2	/image/photo
3	/bird
4	/bird/crane
5	/machine
6	/machine/crane

キーワードがファイルに付与されると，そのファイルの実体へのシンボリックリンクが，キーワードに対応するディレクトリに配置される．このリンクの名前はファイル名，キーワードIDのリストより生成される．例えば，表2の状況において，*/image/photo*, */bird/crane*//*fig.jpg* という一意名を持つファイルについて考える．リンクの名前は `.fig.jpg#2,4#` となり，これを `/image/photo` と `/bird/crane` の両方のディレクトリに配置する．

ファイル指定が与えられると，キーワードに対応するディレクトリがオープンされ，中のリンクが参照される．例えば，`/image/photo//fig.jpg` というファイル指定が与えられた場合，ディレクトリ `/image/photo` がオープンされ，シンボリックリンク `.fig.jpg#2,4#` が参照される．

6.1.3 ファイルのキーワード数の管理

プロトタイプシステムでは，ファイルに対するキーワードが全て削除されると，そのファイルは名前空間から削除され，参照することができなくなる．

参照されなくなったファイルがどのように扱われるかは，ファイルシステムの実装に依存する．例えば，参照されなくなった時点でファイルそのものを自動的に削除することが考えられる．

プロトタイプシステムでは，削除の対象となるかどうかの基準として，付与されたキーワードの数を用いる．キーワードが一つでも付与されているファイルは，ユーザにより参照されているものとして，削除の対象とはならない．ファイルに対するキーワード数の管理のために，ファイルの実体が存在するディレクトリ内に，その実体に対するキーワードの数を保存するテーブルを用意する．表3はそのテーブルの例である．

表 3 キーワード数管理テーブル

ファイル名	キーワード数
`main.tex`	2
`project1.jpg`	10
`tmp.txt`	0

ユーザからのキーワードの付与が行われた時，シンボリックリンクを作成すると共に，キーワード数管理テーブルのキーワード数を1増加する．

ユーザによるキーワードの削除が行われた時，シンボリックリンクを削除すると共にキーワード数管理テーブルのキーワード数を1減少させる．キーワード数が0になったファイルは，ユーザから参照されなくなったことを示す．

6.2 名前空間の統合の実装

6.2.1 名前空間の統合のしくみ

各ユーザのキーワード階層はキーワードテーブルに登録される．キーワードテーブルは，各ユーザ毎に個別のテーブルを用意する方法と，全ユーザのキーワードを一つのテーブルで管理する方法とがある．テーブルを格納する空間の大きさや，テーブルからキーワードを検索する際の性能を考慮すると，ユーザ毎に個別に用意するのが望ましい．

名前空間を統合した際にファイル指定が与えられると，それぞれのユーザのキーワードテーブルを個別に検索し，対応するディレクトリがオープンされ，中のリンクが参照される．例えば，`/image/photo//fig2.jpg` というファ

イル指定が与えられた場合，それぞれのユーザのディレクトリ /image/photo がオープンされ，fig2.jpg を含むシンボリックリンクが参照される．

6.2.2 キーワード読み替えの実装

キーワードの読み替えは，読み替えの対象となるユーザで区別する必要がある．したがって，読み替えを実現するためには，対象となるユーザ名，読み替え対象となるキーワード，読み替えたキーワードの情報が必要となる．

そこで，これらの情報を表4に示すような読み替えテーブルで管理する．この読み替えテーブルはユーザ毎に個別に用意され，いつでも参照，内容の変更を行なうことができる．

表 4　読み替えテーブル

ユーザ名	読み替え前	読み替え後
todoroki	photo	photograph
tada	picture	photograph

読み替えは，対象となるユーザの名前空間を統合した場合に行なわれる．上記のテーブルの場合は，ユーザ *todoroki* の名前空間を統合している際に，ファイル指定に *photograph* が含まれていた場合，読み替えを行う．例えば，/image/photograph というファイル指定が与えられたと仮定する．この場合，読み替えによって */image/photograph* と */image/photo* が同時に指定されたものとして扱われる．

6.2.3 他ユーザのファイルへのキーワード付与機能の実装

他ユーザのファイルへのキーワード付与は，通常のキーワード付与と同様の方法で実現できる．

付与される階層的キーワードに対応するディレクトリ内に，対象となるファイルの実体へのシンボリックリンクを用意し，ファイル名にキーワード ID の情報を付与する．

例えば，ユーザ A が保有する，*/image/photo*, */bird/crane//fig.jpg* という一意名を持つファイルに対して，ユーザ B が，/photograph/jpg, /animal/bird というキーワードを付与することについて考える．キーワード付与の対象となるファイルの実体へのシンボリックリンクを，ユーザ B の */photograph/jpg* と */animal/bird* のディレクトリに配置する．それぞれのキーワードのキーワード ID が，6，8 であったとすると，このリンクの名前は，.fig.jpg#6,8# とする．このように実装することで，従来と同様の方法でファイルを参照することができる．上記の例のようにキーワードを付与した場合，ユーザ B が新たに付与したキーワードと，ユーザ A が付与したキーワードとを同時に指定してこのファイルを探し出すことはできない．これは，名前空間を統合した際には，それぞれのキーワードテーブルを個別に検索するように実装しているためである．ユーザ A が付与したキーワードを用いてこのファイルを参照したい場合には，そのキーワードについてもユーザ B が新たに付与しておく必要がある．

このような実装を行った場合に問題が発生する場合がある．上記のようにしてキーワードを付与されたファイルに対して，それを保有するユーザがキーワードを全て削除した場合である．他ユーザから付与されたキーワードの情報は，それを保有するユーザの名前空間で管理されていないため，ファイルはユーザから参照されなくなったと見なされる．したがって，ファイルシステムによっては，削除の対象となってしまう可能性がある．

プロトタイプシステムでは，付与されているキーワードの数を，キーワード数管理テーブルで保持している．ファイルを保有するユーザと他ユーザのキーワードを平等に扱うために，他ユーザからのキーワード付与が行われた場合，その情報もキーワード数管理テーブルで保持するようにする．したがって，他ユーザがキーワードを付与した場合には，キーワード数管理テーブルのキーワード数を増加させ，逆に削除した場合は，キーワード数を減少させるようにする．

7. あとがき

我々が提案する階層的キーワード名前管理は，従来の名前管理と比較してファイルの共有に適していることを示した．また，ユーザ間で名前空間の共有を行う際に発生する問題点について述べ，この問題に対する解決手法を与えた．我々が実装したプロトタイプシステムについて述べ，ファイルの共有の実現のための手法を与えた．

今後は，名前空間を統合した際に，ファイルの検索を容易にするユーザインターフェースの検討を行う予定である．

参考文献

[1] D. B. Terry, "Distributed Name Servers: Naming and Caching in Large Distributed Computing Environment", Technical Report CSL-85-1, Xerox Palo Alto Research Center, Feb 1985.

[2] S. Sechrest and M. McClennen, "Blending hierarchical and attribute-based file naming", IEEE Proc. of the 12th IEEE Intl. Conf. on Distributed Computing Systems, pp.572-580, Yokohama, Japan, Jun 1992.

[3] D.K. Gifford, P. Jouvelot, M. A. Shedon and J. W. O'Toole, "Semantic File Systems", ACM Proc. of the 13th ACM Symp. on Operating Systems Principles, pp.16-25, Pacific Grove, CA, Oct 1991.

[4] B. C. Neumann, "The Prospero filesystem: A Global File System Based on the Virtual System Model", Proc. USENIX Workshop on File Systems. May, 1992.