国際研究論叢　25（３）：107～121，2012

3DCG を利用してインタラクティブにファイル関係構造を可視化するファイルマネージャの提案

下　條　善　史*

A Proposal of a Three Dimensional File Manager That Can Classifies Directories Interactively

Zenshi Shimojo *

アブストラクト

現在多くのコンピュータシステムで使用されているファイルシステムのほとんどはツリー型のディレクトリ構造を持ち、ユーザはさまざまな業務上で生成される数多くのファイルをツリー構造の上で管理している。しかしツリー型ファイルシステムは OS 上での資源管理には優れていても、ユーザの複雑な業務を表現するにはあまりに貧弱な能力しか持っていないことは、すでにさまざまな方面から指摘されている。本稿は、現行のファイルシステムにおけるディレクトリ構造はそのままに、ディレクトリ名やファイル名に表現されているファイルの分類情報を元にして、ファイルアイコンを３次元空間上に配置し、各次元の振り分けをインタラクティブにユーザが指定することによって、さまざまな局面に適したディレクトリ構造を仮想的に構築し、可視化するシステムの提案である。本システムは現行ファイルシステムの上で動作しファイルシステムには影響を与えずに、業務に必要な作業環境を仮想的に作り出すだけの、いわばファイルシステムの補完をするためのシステムと言う位置づけである。

Abstract

Most of the filesystem used with a lot of computer systems have the tree model directories now, and the user is managing a lot of files generated on various works on the tree model. However, it has already been pointed out that it has only a too poor ability from various districts so that the tree model may express user's complex works even if it is excellent in the resource management on OS. Because the directory structure in a present filesystem arranges the file icon in three dimensions based on classification information on the file expressed in the directory name and the file name, and specifies the distribution of each dimension for the state as it is interactively by the user,

＊しもじょう　ぜんし：大阪国際大学現代社会学部准教授〈2011.12.9受理〉

this text constructs, and is a proposal of the system that makes it to visible virtual as for the directory structure that is appropriate for various phases. This system operates on a present filesystem and is a location of system only of the virtual production of the working environment necessary for the works influencing to interpolate the filesystem so to speak to the filesystem.

キーワード

ファイルシステムブラウザ、業務構造の可視化、３次元 CG、仮想ディレクトリ、ツリー型ディレクトリ

Key words

Filesystem Browser, Visualization of the job structure, 3DCG, Virtual Directory, Directories on the tree model.

１．はじめに

現在一般的に使用されている多くの OS（Windows、UNIX など）ではツリー型の階層構造を持つファイルシステムが使われている。そしてユーザの行うさまざまな局面でのファイル管理も、ツリー構造のディレクトリを基にして行われており、OS が提供するファイルシステムブラウザ（Windows では Explorer、UNIX では ls や df、X 上で動作する各デスクトップ環境上のファイルマネージャなど）もこのディレクトリ構造によって分類されたファイル群を操作単位としている。しかしツリー構造上のファイルを検索するにはファイルの分類を、ファイル配置を行った時と同じ順序でたどらなければならず、ファイル検索を頻繁に行う業務には不向きである。

さらに典型的な個人ユーザはその分類の仕方も一貫しているとは言えず、終始同じ基準によって分類していないことも多い。すなわちコンピュータのユーザは貧弱なファイルシステムの上に、構造的に欠点のあるツリーを作り、その上でファイル操作をしていることになる。別の観点では、一般ユーザは業務のファイルと個人のファイルをさまざまな形でそれぞれの分類基準で整理しているが、その一方、ユーザが利用している市販のアプリケーションプログラムはそれぞれのポリシーでディレクトリを作成し、分類整理しているという現状がある。

個人が利用するコンピュータは近年ますます廉価になり、そのストレージも大容量化の一途をたどっており、個人がその業務や個人的用途で管理するファイルは、その種類も容量も増大の傾向にあると言える。その中でそれらのファイルを整理し、業務や個人利用を効率的に進めるには、現在のファイルシステムは機能的に貧弱で、十分な性能を持っているとは言えない。

このような状況で、ファイルシステム自体の機能を強化する動きもさまざまに行われているが、本稿では、現行のツリー型ファイルシステム上のファイル群を、ファイルシステ

ムに記録されている情報によって改めて分類整理し、３次元空間に再構成してグラフィカルに表示することによってそれぞれの業務に適したファイル管理が行えるようにするシステムの構築を目指した。本システムは現行のファイルシステムのディレクトリ構造には影響を与えず、本来の状態を維持したままそれぞれの業務に特化した仮想的なディレクトリを作成するので、業務の内容に応じてインタラクティブにディレクトリ構造を変化させ、表示することができる。

以下、第２章でツリー型ファイルシステムの抱える問題点とそれに対する様々なアプローチを整理し、第３章で本稿のめざすシステムの概要を示す。第４章では Windows のファイルシステムを例に、現行のファイルシステムで管理されているファイル情報の利用方法を検討し、第５章では、それらの情報を３次元空間に投影するしくみを紹介する。第６章では、本システムから行うファイルの操作方法を紹介し、第７章で仮想的に構築したディレクトリを使って業務を行ってみた実験を紹介する。そして第８章で、本システムの問題点を明らかにし、本システムの限界や適用範囲を定めてみることとする。

２．ディレクトリ構造の理想と現実

現在一般的に利用されているほとんどの OS では、ファイルを管理するファイルシステムにおいてツリー構造のモデルを採用している。そのひとつの理由は、ツリー型モデルは単純でユーザにとってもわかりやすく、しかも大量のファイルを分類整理して格納できるモデルであることによる。しかし一方で、ツリー構造のファイルシステムにはさまざまな欠点があることが指摘されている。たとえば後藤ら[1]の研究では、ツリー構造のファイルシステムは目的のファイルへ検索木をたどって特定する際、構造を作成したときと同じ順序でたどらなければ目的のファイルは検索できない、またいったん構造化されたディレクトリも長年の使用でファイル数が増大するので、再構成をする必要がしばしば発生し、その際に当初の構造が変化するため、検索が困難になる、といったユーザにとっての欠点が指摘されている。

したがってツリー構造のファイルシステムを使用して理想に近い作業をするには、ディレクトリの作成時点から将来の拡張なども予想して、最適なディレクトリ構造を構築しなければならない。掛下[2]では、Halasz、Papazoglou らの挙げた情報分類のための必要条件[3][4]をまとめて、さらに拡張した以下の７項目を挙げている。

（１）データ分類項目を自由に定義できる。
（２）データを随時追加できる。
（３）分類に使用する条件に一貫性がある。
（４）さまざまな形式のデータに対応する。
（５）検索の指針を利用者に提示できる。
（６）分類したデータを容易かつ柔軟に検索できる。
（７）情報分類に要する管理コストが低い。

ここでは、「利用者レベルと管理者レベルでバランスのとれた情報分類方式」ということを含めて７項目を定めているが、本稿で議論の対象にする個人ユーザでは、管理者とユーザは同一人物であり、専門の職務としてファイル管理を行うのではなく、業務の合間に管理を行っているのが現状である。一般のユーザが業務を行う傍らこのような分類法を実行することは困難であり、現実に作成されたディレクトリ構造は、さまざまな欠点を抱えたまま成長せざるを得ない。

また、仮にユーザの意図によって厳格で一貫した基準のもとにツリー構造を作成できたとして、そのようなファイルシステムの上でも、業務の状況によっては現行のファイルシステムとは異なるディレクトリ構造の方が効率的に作業できるケースも少なくない。たとえば図１に示すディレクトリは大学等の講義用に作成している年度ごとに整理された担当科目のディレクトリの例である。各々の科目では、授業に使用するテキストや資料を格納するディレクトリ、授業の課題に使用する資料や制作された課題を格納するディレクトリ、そして学生の成績資料を格納しておくディレクトリがそれぞれ設けてある。通常の学期間では、当該年度の授業が進行しているので、年度のディレクトリの中だけを参照していれば十分であるが、たとえばFD推進のために過去の同じ授業での学生の成績を分析したいなどの要求がある際には、図２のような構造で整理されたディレクトリ構造の方が明らかに利用しやすく、作業を効率的に行うことができる。このようなことからも、現在のようなファイルシステムに管理されたままのディレクトリを単位として業務を行うのではなく、それぞれの業務に都合のよいファイル集合をインタラクティブに定義して、それらのファイルをまとめた単位を仮想

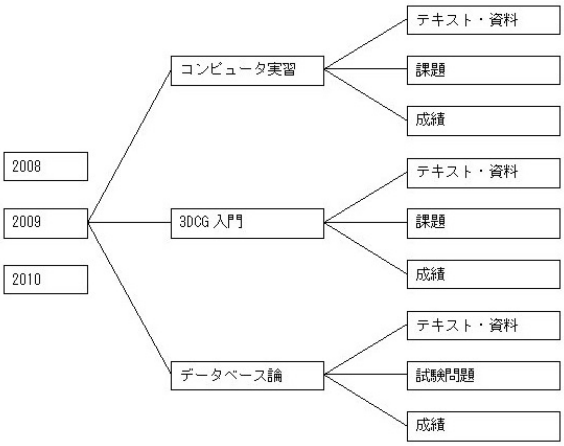

図１　日常業務に適したディレクトリ構造

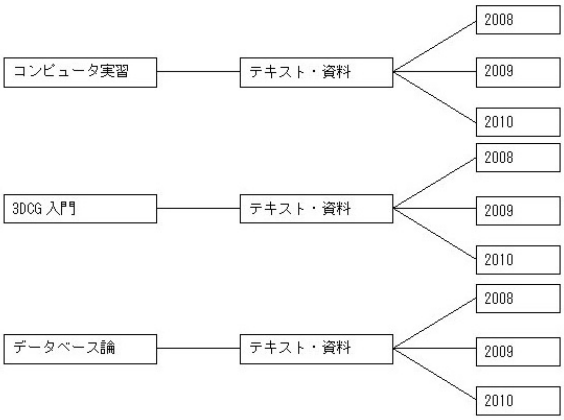

図２　資料整理に適したディレクトリ構造

的なディレクトリとしてユーザに提示し、その環境下で作業を行うのが効率的な作業には重要であると考えられる。逆に、そのようなインタラクティブな仮想ディレクトリの構成が可能となれば、実際に OS で管理されるファイルは、極端に言えばどのような構造のディレクトリ上にあっても業務の効率を落とすことはない。すなわち、すでに不完全なまま成長してしまった現行のファイルシステムはそのまま置くこととして、常に仮想的なディレクトリで作業していけばいいことになる。この際、仮想的なディレクトリ構造をユーザの手によってひとつひとつのファイルについて指示しながら再構成するのでは、新たな作業が発生して効率を逆に落とすこととなる。そこで現行のファイルシステムに記録されているいくつかの情報を使って仮想的なディレクトリ構造を機械的に構築することにする。これによってユーザは、その時点での理想的な作業環境を得ることができるが、作業が終われればファイルシステム上のディレクトリ構造はそのままの状態で維持されており、全体の構造には影響を及ぼさないシステムが構築可能となる。

3．仮想ディレクトリ可視化システム TDF の設計

本提案のシステムは、現状の欠点のあるディレクトリ構造はそのままに、仮想的なディレクトリ構造を３次元空間上に可視化して、さまざまな作業のそれぞれの過程で最適なファイル関係をインタラクティブに構築し、作業環境を提供することを目的としている。現実のファイルシステム上に展開されたディレクトリ構造に対して仮想的なディレクトリ構造を与えて作業を効率化するというアイディアは、たとえば八重樫ら[5]によっても提案されている。八重樫らは、実際のファイルシステム上のツリー構造の他に名前、時間、内容など別個の視点でツリー構造を仮想的に作成することを許したシステムを提案している。しかしこの方法では、別の視点によるツリー構造の作成を自動的に行うことは困難で、ファイルの管理のために新たな作業が発生してしまうことなどが欠点として指摘されている。また別のアイディアとして、既存のディレクトリには必ずしも一貫して記録されていないタスク（作業）というインデックスを新たに設け、タスク毎に新たなディレクトリを別途作成して、そこにそのタスクで使用されたあらゆるファイルのリンクを格納することによって、ファイルの実態とタスクの間の関係を作り上げて、その後のファイル検索の際のインデックスとするというアイディアが、前述の後藤ら[1]によって提案されている。しかし後藤らのシステムでは、そのタスク自体の分類が必要となる事態が想定されていない。現実にはユーザが日々行っているタスクは１日の間でも数件に上るのが普通であり、長期にわたって蓄積していくと、インデックスとなるタスク自体を検索するためのインデックスが必要となる。タスクという、ユーザ経験がインデックスとなりうるという事実はユーザアンケートなどで確認されているので、タスク自体をどうインデックス付けするかの問題を解決することによって、効率的な作業環境をこのアイディアは作成しうると考えられる。

このような過去の研究を参考に、本稿で提案するシステム TDF（Three Dimensional graphical File manager）では、以下のような設計方針を挙げることとした。

● TDF は現行のツリー型ファイルシステムの上で動作するアプリケーションとして作成する。
● TDF はアプリケーションウィンドウ上のグラフィック領域に仮想的な３次元空間を表示し、ファイルを示すアイコンを３つの変量で示される空間上に表示することでファイルの分類を表現する。
● TDF では各ファイルは、仮想空間上の位置、表示されるアイコンの形状、そして色の５次元の情報でファイルの分類を示す。
●３次元空間上へのファイルアイコンの表示のほか、ユーザはアイコンをクリックしてファイルに対するさまざまな操作をすることができる。
● TDF は取り扱うファイルについて、現行のファイルシステム上の位置の変更は行わない。また現行のファイルシステム上のディレクトリの操作は行わない。

TDF でできるファイルに対する操作は以下の通り。

アプリケーションの起動、ファイルのオープン、ファイルの削除、ファイルのリネーム、ファイルのコピー、ファイルの属性変更。

また、TDF は一般的なファイルマネージャでは可能な以下の操作は行わない。これらはすべてディレクトリに関する操作であり、TDF はあくまで仮想的なファイルの操作環境を与えるだけで、現行のファイルシステムのツリー構造を組み替えるものではないという方針による。

ディレクトリの作成、ディレクトリの移動、ディレクトリの削除、ディレクトリのリネーム、ディレクトリの属性変更。

以上の方針で TDF を構成することとする。次章では３次元空間への配置を定める、ファイルの分類方法について述べていく。

４．仮想ディレクトリを作成するためのファイルの属性

現行のファイルシステムに保存された情報を使ってディレクトリ構造を再構成するために、まずは現行のファイルシステムがどのような情報を格納しているかを調べてみる。一般的なパソコンシステムで利用頻度の高い Windows シリーズのファイルシステムである NTFS の管理する主なファイル属性は以下のとおりである[7]。NTFS は IEEE が策定している OS の標準規格である POSIX に準拠したファイルシステムである。その他、主に

サーバー用途に利用されることの多い UNIX/LINUX のファイルシステムも POSIX 準拠であり、同様の項目を原則として保持している。

1．ファイル名
2．パス名
3．所有者名
4．パーミッション
5．作成日時
6．アクセス日時
7．更新日時
8．ファイルサイズ

これらのうち、ユーザが作業を進める際に興味のある項目は、ファイルを識別するための項目（おもに名前）と、データの種類、すなわちファイルフォーマット、データの新しさ（作成・更新日付）、そしてファイルの分類項目である。ファイルフォーマットはファイルシステムに直接保持されるものではないが、多くの場合拡張子という形でファイル名の一部（原則としてファイル名の末尾の数文字）に保管された情報を利用する。ファイルの分類項目は、仮に理想的にディレクトリ構造が構築されているとすれば、分類の基本的原則、すなわち（1）一貫性があり、（2）排他的であり、（3）全体を覆うものであり、（4）かつ飛躍がないものでなければならない[6]。ツリー構造に展開されるディレクトリ上の分類項目は前述の通りこれらの条件をすべて満たすには程遠いものであるが、一つ一つのディレクトリ名は、その先にあるファイルの分類項目を示すものであり、これらの順序性を取り除き、掛下[2]の提案する多次元の独立した複数のツリーによる分類に変換すれば、それぞれのツリーにおいてはこれらの原則が遵守される結果となりうる。具体的にはファイルシステムに保持されるパス名は、そのファイルの分類項目を示すディレクトリ名が区切り記号で連結されたものであり、そこから複数のディレクトリ名を分離して取り出せば、それはファイルの分類項目を一覧したことになる。そしてパス名から分離した時点で分類項目自体の親子関係は削除されるので、第2章で問題となったツリー構造の欠点の一つである、一定の順序でなければ目的のファイルまでたどれない、という点をクリアすることができる。そこでこれらのディレクトリ名はそのファイルの分類を表すキーワードとして扱うことで、ツリー構造に縛られない分類と検索が可能になる。ツリー構造の順序によって妨げられていた分類の一貫性を復元することができるということになる。

これらを含めて TDF が3次元空間に投影することのできるファイル属性は以下のとおりとする。

1．ファイル名
2．ファイル名から分離した拡張子
3．パス名から分離したディレクトリ名

４．ファイルサイズ
５．作成日時
６．アクセス日時
７．更新日時

これらのうち２. の拡張子については、一般に Windows でも UNIX でもファイル末尾のドット（.）以下の数文字（１文字から５～６文字程度、多いのは３文字だが、拡張子がつけられていないファイルもしばしば存在する）で表される文字列であるが、この文字は習慣的にファイルフォーマット、すなわちファイルに格納されるデータの形式を示すこととなっており、これがないとデータフォーマットを認識しないアプリケーションも多く、ファイルの用途を表す文字列として利用できる項目である。また３. のディレクトリ名は、各ファイルにひとつ以上、多くの場合数個程度収集できると思われ、ファイル分類のキーワードとして用いることができる。個人ユーザがファイルをツリー構造で分類する際、一つのファイルに平均いくつ程度のディレクトリ名が付与されているかについて調査した事例は少ないが、経験的には３～４個程度を中心として、最大でも７～８個程度にとどまるものと思われる（OS や大規模なシステムプログラム、ソースプログラムとライブラリの集合などでは十数階層に達するツリーもしばしば現れることがある）。５. から７. の各日時は、多くの OS で情報が保持されているが、アプリケーションによっては正しくこれらの日付を保持しないものも存在し、あるいはユーザの手で変更することもできる。なによりコピーや移動など内容の編集を伴わない操作でも変更されることがあるので、利用できる局面は限られると考えるべきであろう。一方で日付と時刻に含まれる情報はいわばユーザのライフスタイルに直結した情報であり、状況によっては有力な分類項目になることもありうる。たとえば大学の講義に使用されるファイルは、講義が週間単位で行われるとすると、アクセス時刻の曜日の項目は常に一定になるはずであろう。その他季節（月）や時刻に依存する分類項目も考えられよう。いずれにしても、これらの項目のいくつかを選択してユーザは作業に使用するファイル群を仮想ディレクトリに再構成することになる。

５．３次元空間へのファイル属性の投影

ユーザが利用するアプリケーションソフトの多くは、ユーザの作業結果をアプリケーションデータとして保存する際、デフォルトのディレクトリに保存する設定となっている。しかしユーザの意図があれば、他のディレクトリを指定することもできるようになっている。ユーザがディレクトリを指定する場合、そのディレクトリはユーザ自身の分類方法に従って作成されたものであり、ユーザがその分類方法に適した名前を付けている。すなわち、人、もの、場所、時刻、出来事、概念、といったオブジェクトを示す名詞のうちのいくつかがディレクトリ名となるのが一般的である。これらのうちのいくつかの情報は、ディレクトリ名以外のところでもファイルシステム上に保存されている。すなわちファイ

ルの作成・アクセス・更新の時刻やファイルの作成者の情報などは、ディレクトリ名に使われることもあるが、OS が自動的に保存する属性でもある。いずれにしても、最終的にいくつかのディレクトリツリーの先にあるノードとしてファイルを保存するのであれば、そのファイルはそれにいたるすべてのディレクトリ名が表す属性を備えていると考えられる。したがって、そのファイルを他のファイルと識別するための情報として利用できる情報であるといえる。

TDF では、ユーザがこれらの情報のうちのどれを使ってファイルを分類するかを選択することができる。一般的なファイルマネージャであれば、ユーザはまずディレクトリツリー中の任意のディレクトリを作業スペースとして選択し、その中のノードとして存在するファイル群を作業の対象とする。それに対して TDF は、ユーザがこれから行おうとしている作業をいくつかの分類項目の属性として表現し、それに該当するファイル群が３次元空間上のファイルの集まりとして表示される。ここがこれからの作業空間であり、またそこにあるファイル群が操作対象となる。そこに至る手順としては、ユーザが分類項目を選択すれば、後はシステムがこれらのファイルを空間上にアイコンとして表示する。これを空間上のある位置にカメラを据えることで、空間ごとこれらのファイルを概観することができるようになる。仮想的な空間は言うまでもなく３つしか次元を与えられていない。CG アニメーションを利用して４つ目の次元を導入することができるにしても、限られた次元数での分類であるから、分類項目は原則として３つに限られる。そして最もその時の業務にふさわしい３つを選び出すための取捨選択を行わなければならない。表示の際のアイコンの形によってファイルフォーマットを表し、またアイコンの色によって分類項目の一つを表せば、５次元または６次元の分類も条件付きで可能となる。前章で、通常のディレクトリ構造の階層を３から４程度と考えたことを適用すれば、３次元空間に実用上は十分なディレクトリの構造を表現できると考えられる。

第７章ではこのことも含めて評価する実験を行った。

６．TDF のユーザインタフェース

TDF は OS が提供するファイルマネージャと同等の機能を提供するシステムである。したがって単に仮想的なディレクトリ構造を表示するだけでなく、画面上でファイルの操作を実現する必要がある。３章で述べたように TDF はファイルマネージャではありながら、実際のファイルシステム上のディレクトリ構造を変更する操作は行わない。したがって前述のとおり、仮想ディレクトリ構造の表示以外の TDF の機能として実現するのは、（１）アプリケーションの起動、（２）ファイルのオープン、（３）ファイルの削除、（４）ファイルのリネーム、（５）ファイルのコピー、（６）ファイルの属性変更の６種類である。さらにこれらの機能を補助する機能として、（７）ファイルの複数選択、（８）コピー（移動）先の実ディレクトリの表示と選択、の機能も実現しなければならない。現在 TDF は Python 言語を使用して開発中であるが、（１）アプリケーションの起動と（２）ファイルのオープンは OS のシステムコールで簡単に実現できる。（３）～（６）も同様にシステ

ムコールで実現可能だが、（4）ファイルのリネームは新ファイル名の入力が必要となり、（6）ファイルの属性変更は現在の属性と変更後の属性を表示変更するためのサブウィンドウが必要である。TDF の特有の機能として実現に工夫が必要なのは、（7）ファイルの複数選択である。ファイルを一つ一つクリック等で選択していくことは難しいことではない。またマウス等の移動によって画面上の範囲を指定し、その中に入るファイルをすべて選択するというのも実現はさほど困難ではないだろう。問題は、3次元空間に表示するために、カメラの状態によっては他のアイコンの下に隠れるアイコンが存在することである。アイコンの重なりが多重になった際、ファイルの密集具合によっては選択が困難になることもありうる。また正しく選択されていることを確認する画面も必要となるだろう。そのような場面が起こらないように、まず表示の対象となるファイル群をまず絞るべきであるし、そのようなことが起こったとしたら、本来の実ファイルマネージャとの併用も視野に入れる必要がある。（8）コピー先の実ディレクトリの表示と選択も、同様の方法で解決できると考えられる。そして最後に TDF 特有の機能として、一度分類整理した仮想ディレクトリの状態を保存し、ファイルの構成が変化した後で再び同一の仮想ディレクトリを再現することができなければならない。仮想ディレクトリの状態は、1．各軸に変換される項目の選択状況、2．色、アイコンの形状にアサインされた項目の選択状況、3．カメラの視点、向き、視野角、のそれぞれの情報である。これらの情報は WWW ブラウザのように、ブックマークのようなものを用意して手軽に呼び出せるようにすべきであろうが、それは後の拡張にゆだねることにする。

7．評価実験

ここでは、実際にユーザが数年間にわたって自宅のパソコンシステム上に作成した2例のディレクトリ構造を TDF 上にそれぞれ投影し、仮想ディレクトリを作成して概観した感想を述べる。実際の評価実験はより多くのケースを基に行うべきである。

最初のケースは筆者個人のユーザディレクトリをこのシステムによって投影してみた結果である。まずは、Windows 7上に構築されたディレクトリ（Windows のシリーズではディレクトリのことを「フォルダ」と呼んでいるが、ここでは「ディレクトリ」で統一する）で構成されるツリー型構造の一部を表したものである。

図3は市販の3DCG プロダクションツールである SHADE 12によって作成した、3次元空間に投影されたファイル群の一部である。分割された4つの画面は右上が全体の透視図であり、左上が上面図、左下が正面図、そして右下が右面図となっている。またこの図は左右方向を x 軸、上下方向を y 軸、前後方向を z 軸とし、x 軸にはファイルの拡張子で表わされるファイルフォーマット、y 軸はファイル名でソートしたもの、そして z 軸は作成時刻となっている。

この例の場合、3軸にディレクトリ名に現れる分類項目は使用していない。したがってユーザがツリー構造の中に分類した際の分類基準はここには表れていない。しかし逆に

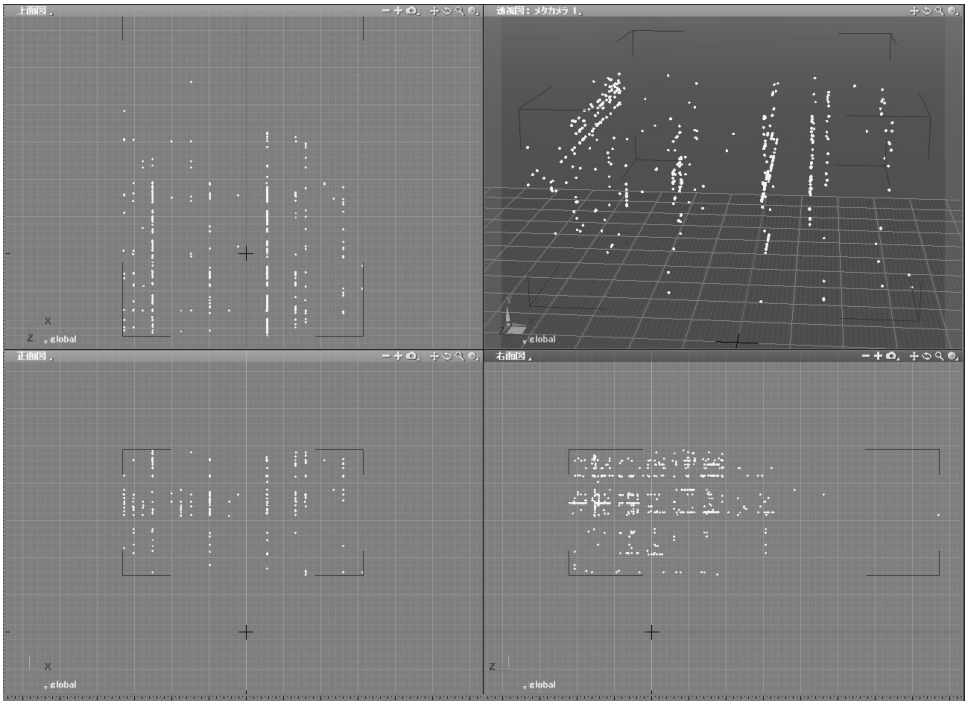

図 3　個人ディレクトリの 3D 展開

ユーザの分類の意図にはもともとなかった構造が分類できている部分がある。たとえば右面図を見ると、横方向にファイルのタイムスタンプが配置されているので、これは時間軸を表す軸である。このファイル群は 3 年間の科目で使用したファイルを集めているので、途中ファイルの作成されていない期間がファイルの存在しない空間としてファイルアイコンの間に存在し、ファイル群を分けている。さらに良く見ると、夏休み、春休みの期間がはっきり見てとれて、ユーザの分類にない「前期・後期」の分類が表れていることがわかる。他にも正面図から見れば、左右方向にファイルの拡張子が表すファイルフォーマットの分類を示していることになる。ファイルフォーマットは他の項目に比べても値の種類が比較的少なく、したがって分類も明らかに行われている。この図の場合は左から数えて最初の一群は演習課題で提出された学生の作品の一部である JPEG 画像ファイル群（拡張子 JPG）であるし、2 番目の大きなファイル群は各科目で毎回の講義に用意されたパワーポイント（拡張子 PPTX）ファイルの集まりである。このようにディレクトリをまたがったファイルでも、なんらかの共通項があればまとめて選択し、処理することができるのが明らかにわかる。

もうひとつの例は、筆者が本研究を始めたきっかけとなる事例である。図 4 で表わされたファイル群は筆者の父親のパソコンからコピーしたファイル群である。父は経済学の研究者で、長年にわたってさまざまな研究資料をパソコン上に残してきたが、それらが整理

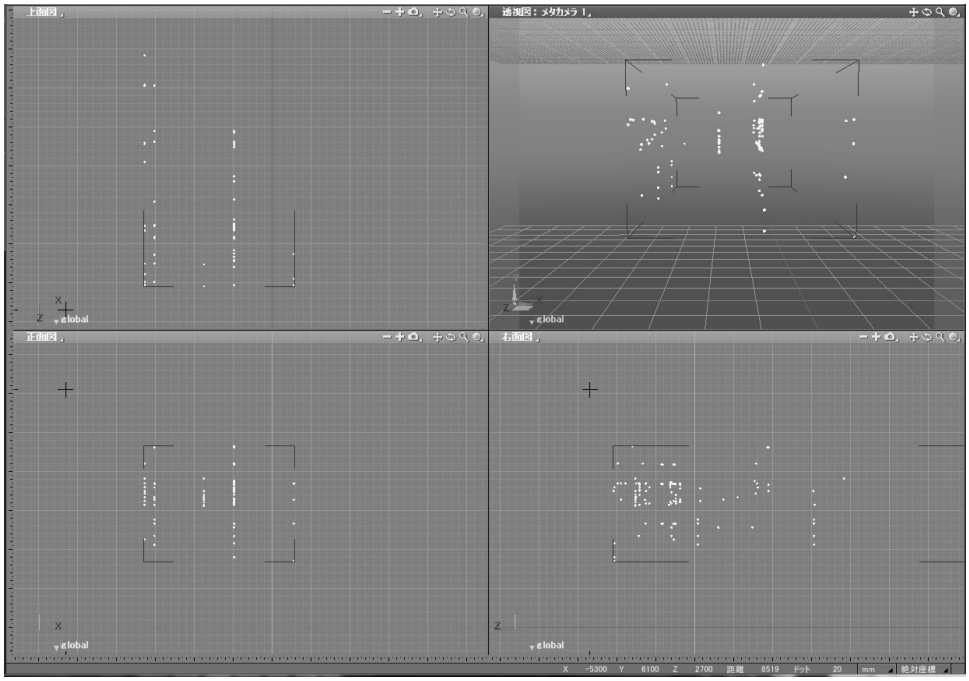

図４　父の遺したファイル群

されぬまま一昨年他界した。遺族として父の業績を紙の資料からだけでなく、パソコンの中からも取り出そうと試みたが、父のファイル群は多数のデバイス上に分散されて残されており、しかもそれらの中に同名、同タイムスタンプのファイルが数多く見られ、全体としてどのような構造で整理されているかが他人にとっては判然としない状態であった。そこで、本稿のようなシステムを考案し、父の残した膨大な資料を整理しようと試みることになったというわけである。

実際の整理はまだ途上であるが、前図のような同じ名前のディレクトリやファイル群の重複はほぼ整理できることがわかった。さらにこの例ではディレクトリ名で表わされる分類項目も分類に利用している。項目の選択によっては仕事上の関連の深さが３次元空間上の位置関係として表わしうることも分かった。

８．本システムの問題点と適用範囲

TDF はあくまでも、現行のファイルシステムと欠点のあるディレクトリ構造を維持したまま、可能な限り作業に特化した環境を作るためのシステムをめざして考案されている。これまでの議論でも何度か登場したように、本来このような機能は根源的には OS 上のファイルシステムの機能として実現するべきものである。外部アプリケーションとして動作する本システムは、効率の点でも、また速度の点でも、明らかに劣ることが予想され

る。各 OS でも、たとえばリレーショナルモデルを適用したファイルシステムの考案や実装が進められており、時間の問題で本システムで実現するような機能はすべての OS で実装されることになると思われる。したがって本システムの意義は、それまでの期間を埋めるだけのものであることを承知している。

このことを前提にしても、本システムにはいくつかの問題点が存在している。解決可能な部分のアイディアも合わせて以下に紹介する。

８．１．分類項目の構造化の方法

本システムの最大の難点は、ディレクトリ名で表現されている各ファイルの分類項目をカテゴライズする方法論である。本システムは仮想ディレクトリの作成を自動化することによって、新たな管理コストをかけることなくディレクトリの再構成を実現しているが、現実問題としてディレクトリ構造に表れる分類項目名は、ユーザが自由につけられるものであり、機械的分類がしにくいケースもありうる。たとえば、業務の時刻に関するディレクトリ名だけでも、「2010」「２０１０」「2010年度」「平成22年」「第15期」というような呼び方が可能であり、これらを統合する辞書のような役割を持つ仕組みを組み込まなければ、自動化はできない。部署名や業務名などの分類名は、ともすると固有名詞であることも考えられ、管理コストをかけずに簡易に拡張できる辞書が必然となる。

８．２．分類結果の検証方法

３次元空間上にファイルアイコンが配置されることで、ユーザが指定したファイル分類の成果となるわけであるが、この結果がユーザの意図を正しく反映しているかどうかを検証する方法とそのコストが問題となる。方法としては、たとえば画面上で選択したファイル群の実ファイルシステム上のフルパス名を別ウィンドウなどで表示してやる方法などが考えられるが、それが意図どおりの結果であることを確認するためには、すべてのファイル名を目視でチェックするしかない。仮に操作の過程でヒューマンエラーに起因する選択間違いなどがあったとしても、システム側からはそれをチェックする方法がない。この状況で可能なのは、まずはシステム内での操作に対してログを完備すること。そして実ファイルシステムに影響する操作（コピーやリネーム、特に削除など）にはシステムを終了した後でもキャンセルやリカバーができるようにしておくことであろう。

８．３．３次元空間の操作方法

３次元空間に表示されたアイコンは、カメラの向きや視野角によっては重なって表示されることがしばしばあり、それを視認上、かつ操作上も分離できなければならない。カメラの向きを変えて容易に分離できる状況であればさしたる問題ではないが、アイコンが密集した場合には解決が困難な場合もありうる。また３次元空間を画面という２次元平面に投影した際のアイコン同士の見かけの距離と、３次元空間での距離は必ずしも一致しないが、２次元の情報しか与えられないマウスの操作では画面に対する奥行き方向での操作方法が限定されるので、満足に操作できない状況も考えられる。これらの問題については３

次元入力デバイスと立体視システムの導入などが考えられるものの、このシステムがそもそも志向する、現在の状況への緊急避難的な対応という方向性の上では現実的ではない。

８．４．処理速度の問題

３次元空間を画面上に表示するという点に関しては、昨今の個人のコンピューティング環境の進歩から、さほど危惧するところはないが、このシステムでボトルネックとなるのは、大量のファイル情報を OS の管理するファイルシステムから読み込んでこなければならないという部分である。現在のパーソナルコンピュータはグラフィックス性能や CPU の処理能力は格段に進歩しており、ネットワークアクセスとハードディスクアクセスにボトルネックが移動している。本システムは場合によっては複数のハードディスクから大量のファイル情報をロードしなければならず、表示対象を増やすほど、初期動作は遅くなってしまう。これは利用する側ができるだけ範囲を限定して利用することを心がけるしか解決方法がない。

以上が現時点での本システムの明らかな問題点である。これらのいくつかは使用方法で解決することもできるので、それがすなわち本システムの適用範囲を定めることになる。

９．おわりに

本システムの検討に際して基本的に筆者の心にあったのは、すでにさまざまな場所で言及されている「ツリー型モデルに基づくファイルシステムへの不満」であった。筆者のパーソナルコンピュータ上のファイルシステムに配置されたディレクトリ構造は、いわば筆者の思考過程であり、またその蓄積である。７章で登場した父の例のように、筆者自身の業績もこのファイルシステムから取り出されることもあるかもしれないとすれば、せめてもう少しなりとも整然とした構造であるようにと、長年ディレクトリ構造の作り方には試行錯誤を繰り返してきたものである。一方、コンピュータシステムを当たり前のように使って業務を行ってきた経験からは、本システムの基本的な考え方である「理想的なディレクトリ構造は、状況と業務の内容によって異なる」も得られた。したがって恒常的なディレクトリ構造は、業務の一側面を表現はしても、すべての業務で常に理想的である構造は存在しないことになる。これらの思考過程から、本システムの用途は決して理想的なディレクトリ構造を実現することにあるのではなく、仮想的で柔軟なディレクトリを状況に応じて提供するシステムとしての実現を目指すことになったという次第である。このような考え方に基づくファイルマネージャは、データの集合にさまざまな操作を加えてデータの実態を明らかにする表計算ソフトのように、ファイルの集合を前に、さまざまな構造をインタラクティブに当てはめることによって、ファイル群が表現する業務の構造や、業務の展開に対する新しいアイディアを現出するツールとなることを期待して本稿の結びとすることにする。

参考文献

［1］後藤啓太、ほか：ユーザ経験としてのタスク－資源関係に基づくファイル管理手法、情報処理学会研究報告 IPSJ SIG Technical Report 2008-HCI-129、2008。
［2］掛下哲郎、原槙稔幸：多次元分類：木構造とキーワード分類の複合的アプローチ、情報処理学会論文誌（データベース）、vol. 42、No. SIG1（TOD8）、2001年。
［3］Halasz, F.: Reflections on NoteCards: Seven issues for the next generation of hypermedia Systems, Comm, ACM, Vol. 31, No. 7, pp. 836–852 (1988).
［4］Papazoglou, M.P. and Milliner, S.: Pro-active information elicitation in wide-area information networks, Proc. Int. Symp. on Cooperative Database Systems for Advanced Applications (CODAS'96), Kyoto Japan, pp. 2–11 (1996).
［5］八重樫尚彦、白井靖人：情報処理学会研究報告 IPSJ SIG Technical Report 2004-DD-47、2004。
［6］長尾真：知識と推論、岩波講座ソフトウェア科学14、岩波書店（1988）。
［7］http://ja.wikipedia.org/wiki/ ファイルシステム #NTFS