電子文書の長期保管のための枠組: POT

伊戸川 暁† 川合 慧† 山口和紀†

現在の電子情報の管理のあり方は，即時的な情報の共有については研究が比較的進んでおり運用上の実績もあるが，長期的な保存に関しては，様々な問題を抱えたままである．文書自体を支える媒体の寿命が不十分であることもさることながら，文書に付随する管理状態，恒久的な指示方法，管理者の情報などの整備も不十分である．

本論文では，これらの問題を解決するために，POT(Persistent On-line Text) 文書モデル及び POT アーキテクチャを提案する．POT 文書モデルは，百年単位で使用できることと後世の検証に耐えることを目的とした，文書の公開・廃棄制御・履歴・認証・解釈情報のモデルであり，POT アーキテクチャは，各種の技術的革新の要素を漸進的に取り込むことを可能とする超長期電子文書管理システムのための層化アーキテクチャである．これらのモデルは，管理情報もまた一般の文書と同様に扱うことによってメタデータの寿命を保証し，また，ネットワークを通じて情報を共有することによってスタンドアロンのシステムよりも高い信頼性をシステムに与える．

この文書モデルとアーキテクチャの妥当性を検証するために，保管文書に対して想定される操作の机上シミュレーションを行った．さらに，実装の可能性について，現在利用できる技術的要素のサーベイを行なった．

An Architecture for Archiving Electronic Documents: POT

ITOGAWA AKIRA , KAWAI SATORU
and YAMAGUCHI KAZUNORI

Compared with the remarkable advancement on short-term electronic information sharing, long-term information management still remains in its infancy state. For a reliable long-term information management system, we must develop functions such as media lifetime extension, complex metadata management, stable document naming scheme, and arbitration on generations of managers. And the most crucial is a framework in which these functions are integrated in a coherent manner. For such an integrated framework, we propose POT (Persistent On-line Text) Document Model and POT Architecture in this paper.

POT Document Model is designed for organizing documents for scheduled disclosure and disposal, journaling, user authentication, document certification, document format interpretation. Note that documents in this model are supposed to exist for centuries or over, and the future review of the executed document control is expected. POT Architecture is a layered architecture for very-long term electronic document management with the capability to incorporate new techniques in an incremental manner. In POT, the disclosure of such metadata is guaranteed because the management mechanism is uniform over documents and their metadata.

Execution trace for a few expected operations on POT by hand showed that POT framework is promising. And surveys on supporting technologies presently available showed that the feasibility of POT is high.

1. はじめに

現今の情報システムは，即時的・一時的な情報の交換・共有については大きな成果をあげている．その結果，我々が有している情報の多くは既に電子的媒体に記録されている．これらの媒体に記録された情報は，何の痕跡も残さず容易に消去され得るし，消去されなくても，媒体が稼働状態から外されると，記録された情報の再現が急激に難しくなるという特徴をもつ．ところが，Association of Records Managers and Administrators (ARMA) 元会長の David O. Stephens が 10) において指摘しているように，電子情報の長期保管については，あまり配慮がされてこなかった．

作家の私的な往復書簡が後年発見されて作家研究に

† 東京大学大学院 総合文化研究科 広域科学専攻 広域システム科学系
Department of General Systems Studies, Graduate School of Arts and Sciences, The University of Tokyo

大きく寄与したり，江戸時代の庶民のメモがその時代の経済を理解する上で役立っているといった事実に鑑みたとき，電子媒体に記録されている情報がその特徴により完全に消滅してしまうのだとすれば，人類の記憶にとってまことに大きな損失であると言えるだろう．電子情報の世界における文書館の設立は，現代の急務ではないだろうか．

本論文では，電子情報化された時代の記録を将来に留めるために，電子的に蓄積された情報を数百年のスパンで保持するための枠組を提案する．

2. 問題群

本節では，電子文書を長期保管する場合の問題について概観する．

2.1 媒体の寿命

現行の媒体の寿命は非常に限られている．最も普通に使われている媒体であるハードディスクは磁気的に記憶を行っているため，寿命は数年から十数年である．

CD-ROM や CD-R は，磁気的な記録をしていないのでより長く寿命が取れると期待されているが，媒体の出現から十数年しか経っていないこともあり，公称されている寿命についての確証が得られているわけではない．

2.2 命名体系

従来の紙でできた文書の場合，公開時の名前が（普通は）そのまま文書名として使われ，それが廃れることはない．ところが電子文書の配布手段として最も広く用いられている HTTP[3] の場合，文書の参照はファイルの物理的所在を指示することで行っており，物理的所在が変化すると参照が無効になってしまうという問題がある．これは，紙の世界で例えて言えば，書物を書名ではなく，図書館の書架の位置のみで指定しているようなものであり，極めて脆弱である．

2.3 文書のファイル形式

文書自体の長期保存ができればそれで良いかといえば決してそうではなく，文書そのもの以外の要因でも，容易に電子文書は利用できなくなる．

例えば，ワードプロセッサで作成した文書は，それ自身が残っていても，ワードプロセッサのソフトウェアが存在しなくなってしまうと，表示したり印刷したりすることができなくなる．従って，文書を有効に保管するためには，文書を記録したファイルを利用するために必要な，文書のファイル形式などへの情報へのアクセスが保証されなければならない．

2.4 価値の判断

この世に生成される全ての電子情報を永久に保存することはコスト的に不可能であるし，有益な情報を凡庸な情報の山の中に埋もれさせることにもなる．電子情報の量は特に膨大なので，有益な情報を掘り当てるにために大変な労力が必要となってしまう．

従って，文書情報の登録の際に，文書の取捨選択を指定し，また，登録されるものに対して保存期間を設定できる機構が必要である．

2.5 セキュリティ

電子文書は変更が容易であるので，電子文書を故意の改竄（著者によるものを含む）などから守るための手段を講じなければならない．

また，完全に公開されていない文書に対しては，それを流出から保護する手段が必要である．

2.6 管理組織

人間の寿命を越えて文書を保管するためには，文書の管理者権限を，人間を入れ換えつつ数百年にわたって使用し続けることが可能でなければならない．このためには，永続的な識別名が文書の管理者に対しても与えられ，文書の公開・管理記録が引き継がれていく仕組が必要である．

また，保存されるであろう電子文書の分量は広範囲かつ大量になるため，少数のサイトが文書群を集権的に管理することは困難であることが予想される．従って，各所に多数の管理者権限を置いて分散的に管理する体制を整える必要がある．

2.7 メタデータの標準形式の欠如

永続的に電子情報を管理するためには，文書本体の他にも様々な補助情報が必要になってくる．このような補助情報を，元の文書の**メタデータ**という．書誌管理用には Dublin Core[13]，Web 情報のアクセス制限のためには PICS[5] のようなメタデータの標準形式が既に議論されているが，電子文書の長期保存のためのメタデータについては，未だ本格的な議論は始まっていない．

3. POT 文書モデル

前節で述べた様々な問題の解決方法を位置づけるために，本節と次節では，電子文書長期保管のための枠組である**POT (Persistent On-line Text) 文書モデル**及び**POT アーキテクチャ**を導入する．

まず，POT 文書モデルについて詳説する．

3.1 文書とその種別

POT 文書モデルにおける**文書**には，以下の種別がある．

普通文書 POT が保管する一般の文書．

仕様書 文書のファイル形式を記載した文書．本節

参照.

制御文書 文書の管理計画を記載した文書. 4.4 節参照.

認証文書 文書の管理者の認証情報を記した文書. 4.5 節参照.

文書の種別は，リンクの種類によって特定される.

リンクとは，文書間の関係である. 文書層はリンク関係を適切に維持する. そのために，上位の層からリンク構造を崩壊させるような削除が指示された場合，文書層はその削除指示を拒否する.

仕様書・制御文書・認証文書をまとめて**管理文書**と呼び，管理文書が管理する対象の文書を，管理文書の**親文書**と呼ぶことにする. 管理文書を設けることは，2.7 節で挙げたメタデータの問題に一つの解決策を与えるものである. 文書はその管理情報を本体とは別の文書として持ち，文書本体と管理文書とはリンクで結合される. 管理文書は親文書の外部に置かれ，一般の文書と同様に所在から独立な識別名を与えられ，永続的な保管を受ける. これによって管理組織の永続性を保証し，2.6 節の問題を解決する.

3.2 文書の構造

文書識別子が D である文書は，本文と文書の状態の変更履歴より構成される.

本文は，保管されるべき文書の本文のことであり，本文のファイル形式は，文書にリンクされた仕様書によって示される. 本文の可塑性は，文書の種類が制御文書か認証文書であるとき真，普通文書か仕様書であるとき偽である.

履歴は，文書の登録・管理者の変更・制御情報の変更などの管理情報の変更の履歴であって，廃棄のときまで，可塑性は真である. 文書の管理者も直接履歴を操作することはできない. 履歴の存在によって，文書の不審な挙動の検証が可能になる.

本文及び履歴以外の文書の構成要素を，一括して**外枠情報**と呼ぶ. 外枠情報には以下のようなものがある.

- 文書識別子 (D)
- 仕様書へのリンク (D_{form})
- 制御文書へのリンク (D_{ctrl})
- 認証文書へのリンク (D_{cert})

以下，文書 D の本文・履歴・外枠情報を，それぞれ D_{text}, D_{hist}, D_{frame} と表現する.

3.3 管理文書

3.3.1 仕様書

本文の**仕様書**は，のちに忘れられるかもしれないファイル形式の読み方を将来の読者に対して指示する文書である. 文書を POT に登録しようとするには，

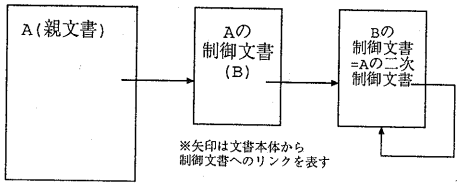


図 1 制御情報の自己言及的構造

Fig. 1 A reflectional structure of the control documentof a control document.

本文が用いているファイル形式の仕様書は，完全に公開されているか，公開される予定がなければならず，削除が指示されるものであってもならない.

このように，仕様書を独立の文書として保管し，文書本体にファイル形式の仕様書を付随させることで，2.3 節の問題を解決する.

3.3.2 制御文書

1 つの POT 文書は，1 個の**制御文書**を持つ. 複数の文書が 1 つの制御文書を共有してもよい.

ある文書 D の制御情報（制御文書 D_{ctrl} の本文 $(D_{ctrl})_{text}$（以下 $\langle D_{ctrl} \rangle$ と略記））は，以下の要素より成る.

重要度 (D^{val}) 将来，親文書 D に期待される歴史的価値を表現したもの. 分散層（後述）以下で，D を構成するファイルの扱いに影響を及ぼす.

公開計画 (D^{dist}_{part}) 親文書の各要素を開示する範囲とそのスケジュール（経時変化）. ここで，$part$ は D の要素名である. 要素ごとの指定なので，本文は見せないまま，履歴のみ見せるというような指定も可能である.

保存年限 (D^{term}) 文書を保持する期間. 廃棄しない文書には「永久」を指定する.

制御文書の管理文書について，以下のように規定する.

- 制御権に関するトラブルを未然に防ぐため，$(D_{ctrl})_{cert}$ は D_{cert} に等しくなければならないものとする. これは，文書管理者のみが制御文書を設定・変更できることを意味する.
- 制御文書のネストについては，2 段までは存在を許すことにする. これは，例えば，制御文書に対して親文書とは異なる公開範囲を定め，一部の人に対しては制御計画のみを明らかにする，ということを可能にするためである. 3 段以上の制御文書のネストは必要性がないため，$((D_{ctrl})_{ctrl})_{ctrl}$ は $(D_{ctrl})_{ctrl}$ に等しくなければならないものとして制御文書の無用な無限退行を防ぐ (図 1).

これらの条件の基で，各文書の管理者は，制御情報と

うしが矛盾しない範囲で自由に制御情報を設定することができる．しかし，制御情報に基づいた文書の状態の履歴は各文書の履歴に記録され，文書削除ののちも保存されるので，非常識な制御情報の設定に対してシステムが一定の抑止力を持つことが期待できる．

3.3.3 認証文書

認証文書は，POTの利用者を特定するのに必要な**認証情報**を記載した文書であり，利用者と1対1に対応する．

文書Dの認証文書 (D_{cert}) で認証される利用者を，その文書の**管理者**と呼び，$owner(D)$ で表す．複数の文書の管理者が同一であることはあり得る．また，親文書の存在しない認証文書も存在し得る．

文書Dの認証文書Pが有する認証情報は，以下の要素から成る．

鍵 (P_{key}) 認証の際に利用者を識別する手段．以下，D_{cert} で認証される利用者を $person(D_{cert})$ と表す．

暗号方式 (P_{method}) 鍵の暗号方式を記した仕様書へのリンク．POTは認証文書を置き換えることによって暗号方式を指定できるので，文書ごとに暗号方式を更新することが可能である．

連絡先 ($D^{contact}$) 電子メールアドレスなど，当該の人物への連絡手段．

認証文書の管理情報は，以下のようになっている．

- $(D_{ctrl})_{form}$ には，暗号方式の仕様書ではなく，D_{ctrl} 自体の書式を記した仕様書へのリンクが入る．
- $(D_{cert})_{ctrl}$ は D_{ctrl} と異なる文書とすることが普通である．これは，D_{cert} が暗号鍵などの重大な個人情報を含んでいるためである．
- $(D_{cert})_{cert}$ に D_{cert} よりも上位の管理者を設定することで，$person(D_{cert}$ が自分の都合の良いように個人情報を改竄することを防ぐことができる．

4. POTアーキテクチャ

POTアーキテクチャでは，長期電子文書管理の機能を5層に整理する (**図2**)．この図では，上位層と下位層の対応する要素の関係を，矢印のない破線で示している．

以下，下位の層より順に，各層の機能を説明する．各層の実現可能性については第6節で述べる．

なお，以下では，一まとまりの計算機群を「**サイト**」と呼び，サイト同士はネットワークを経由して互いに接続しているものとする．

4.1 物理層

各サイトで，**ファイル**（文書を幾つかに分割した単位）を永続的に存在させるための処理を行う層である．媒体の違いも，この層で吸収する．

この層では，記録されたファイルを，サイト内の媒体間を移動したり複製を作成したりすることによって永続化する．但し，滅失の可能性を完全に零にすることはできないので，制御層から与えられる重要度，デバイスの残り容量や平均故障間隔などに従ってファイルを管理し，滅失確率のコントロールを行うように努める．

物理層はサイト毎に保持されるファイルの集積を持ち，1つのファイルは以下の要素より構成される（括弧内は第5節で用いる記号を表す）．

- ファイルの識別子 (f)
- ファイルの内容 (f_c)
- ファイルの重要度 (f_i)

以下，識別子が f であるファイルを $\langle f \rangle$ と表記する．

物理層が行う演算は以下の通りである．なお，矢印の先は，演算の返り値を表す．

$put(f_c, f_i) \rightarrow f$ ファイルを加える．識別子 f は物理層が自動的に生成する．

$get(f) \rightarrow \langle f \rangle$ ファイルを取り出す．

$update(f, f'_c)$ ファイルの内容を f'_c に更新する．

$delete(f)$ ファイルを削除する．

4.2 分散層

分散層は，複数のサイトが協力してファイルを永続化させるための管理を行う．例えば，複数のサイトにファイルの複製を作り，局地的な破壊によってファイルが失われる危険を減らす．このようにして，POTでは，2.1節でふれた媒体の寿命の問題を物理層と分散層の2層で解決する．

複製の数量や配置については，制御層より与えられる重要度，及び制御層や文書層から与えられるファイルの**可塑性**（ファイルの内容が更新される可能性），サイトごとの記憶媒体の状況，サイト間の通信線の容量や安定性などから判断する．

この層より上位の層では，ファイルの物理的な所在とは独立な「位置独立ファイル識別子」によってファイルにアクセスすることができ，その結果，どのサイトからでも等しいサービスが受けられるようになる．これによって，2.2節に挙げた命名体系の問題が解決されることになる．

分散層におけるデータは，**位置独立ファイル**の集積である．位置独立ファイル識別子の等しいファイルは，物理層における実体が複数あっても，分散層では1つ

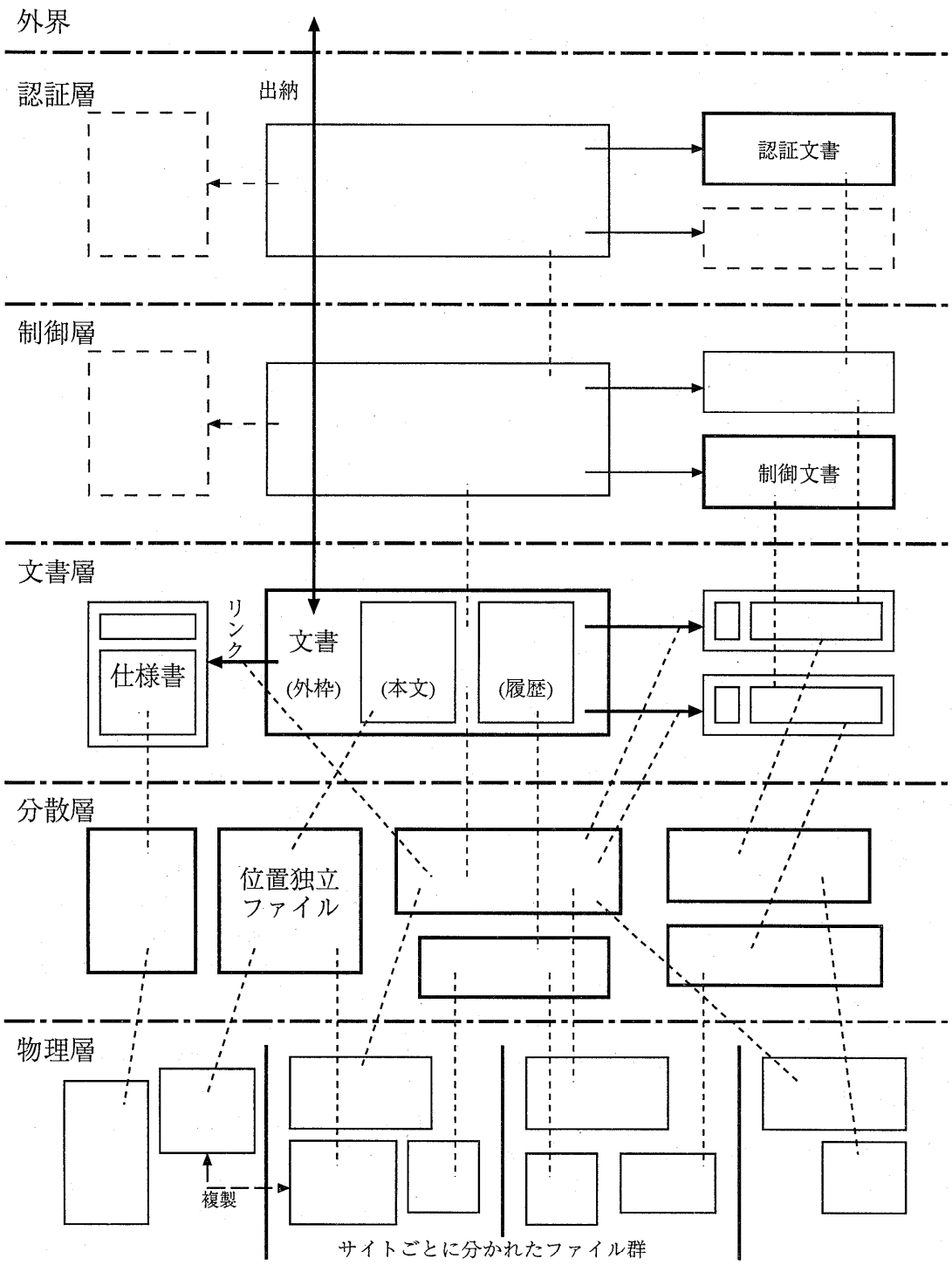


図 2 POT アーキテクチャにおける層関係の概念図
Fig. 2 Layers in POT Architecture.

のファイルとして認識される．

1つの位置独立ファイルは以下の要素から成る．

- 位置独立ファイル識別子 (F)
- 位置独立ファイルの内容 (F_c)
- 位置独立ファイルの重要度 (F_i)
- 位置独立ファイルの可塑性 ($F_p \in boolean$)

F 及び F_p は，物理層のファイルでは本文中 (f_c) に記録されるが，分散層において解釈され，位置独立ファイルの構成要素として扱われる．また，f_c から F と F_p を除いたものが，分散層におけるファイルの内容 (F_c) となる．

可塑性は，ファイルが含まれる文書の種別，文書中のファイルの位置，制御文書から判断できる文書の状態から一意かつ自動的に決まる．

以下，識別子がFである位置独立ファイルを $\langle F \rangle$ のように表記し，単に「ファイル」と述べた場合には位置独立ファイルを指すものとする．

分散層が行う演算は以下の通りである．

$put(F_c, F_i) \rightarrow F$ ファイルを加える．識別子 F は分散層が自動的に生成する．

$get(F) \rightarrow \langle F \rangle$ ファイルを取り出す．

$delete(F)$ ファイル F を削除する．

$update(F, F'_c)$ ファイル F の内容を F'_c に更新する．

$chwrit(F, b)$ ファイル F の可塑性を b に変更する．これは F_p が真の場合のみ実行可能である．

これらの演算は，対応する物理層の演算を呼び出すことで実行する．F と f の対応は分散層がつけるが，対応関係は外部からは隠蔽される．

4.3 文書層

文書層は，分散層でファイルの集積として記録された情報を，文書の集まり及び文書間のリンクの集合として利用者に提示する層である．これによって，分散層ではファイルの集合にすぎなかったものが，リンクで結ばれた文書の集合として見えるようになる．この層の文書は，所在に独立な文書識別子によって指定される．

文書層は，一つの文書を，本文・履歴・外枠の3つの部分に分離し，各々を独立のファイルとして分散層に保存する．

文書層で行う演算は以下の通りである．

$create(D_{text}, D_{form}, D_{ctrl}, D_{cert}, n) \rightarrow D$ 与えられた新たな文書を作成し，識別子 D を作成して返す．ここで，n はDの重要度である．履歴は，文書層が自動的に生成する．

$getframe(D) \rightarrow D_{frame}$ 識別子からDの外枠ファイルのみを取得する．

$get(D, part) \rightarrow D_i$ 識別子と指定された要素 ($part = ctrl or cert or form$) から文書Dの一部を取得する．

$updatelink(D, sort, L)$ 指定された種類 ($sort = ctrl or cert$☆) のリンクをLへのリンクに更新する．

$delete(D)$ 文書Dを削除する．但し，文書が存在したことを記録に残すために，履歴と文書識別子は削除しない．また，削除しようとする文書が他所からリンクされている場合，文書層はその文書の削除を行わない．

4.4 制御層

制御層は，制御文書の有する情報（**制御情報**）に基づいて，文書の開示や処分を決定する層である．例えば，認証層から与えられた利用者の所在地と公開範囲を照合して文書の公開/非公開を決定したり，公開計画や保存年限に従って文書の処分を決定したりする．この層によって，2.4節で述べた問題が解決される．

4.4.1 エージェント

制御文書に定められた日時に文書の削除を行うために，制御層に利用者の代理（エージェント）を置く．

4.4.2 演算

制御層は，文書層での演算に加えて，以下の演算を提供する．

$createctrl(\langle D_{ctrl} \rangle \rightarrow D_{ctrl})$ 制御文書を作成する．$createctrl()$ で作成した制御文書は，後からリンクして用いることができる．

$setctrl(D, D'_{ctrl})$ 文書Dの制御情報へのリンクを D'_{ctrl} に変更する．

$getctrl(D) \rightarrow \langle D_{ctrl} \rangle$ 親文書Dの識別子から制御情報を獲得する．

$canshow(P, X) \rightarrow boolean$ 利用者 P と文書 X の情報から，X の各部分について開示の可否を決定する．

4.5 認証層

認証層はシステムの外側と直接接している層である．この層で利用者を認証して適切な操作権限を与えることによって2.5節の問題を解決する．

認証層の行う演算は以下の通りである．

$createcert(P, Cert)$ 認証情報 $Cert$ の利用者 P を作成する．

$setcert(P, Cert)$ 利用者 P の認証情報を $Cert$ に変更する．

$getcert(P) \rightarrow D_{cert}$ 利用者 $person(P)$ に関する認証情報が記されている認証文書の文書名を取得する．

☆ D_{form} が変更されることはない．

$authexec(P, comm)$ 下位の層に対して，$comm$ を利用者 P の資格で実行する．

$sendmessage(P, message)$ 利用者 P にメッセージ $message$ を送付する．

5. 操 作 例

この節では，具体的な事例において各層がどのように働くかを示すことによって，POT の妥当性を示す．

5.1 文書の閲覧

利用者 A が POT にアクセスして文書 X を読もうとするときに起こることを時間順に説明する．

(1) 認証層は，A の認証文書を文書層から取り出し，その情報に基づいて A を認証する．

(2) 認証層は，制御層の $canshow(A, X)$ により，文書 X が利用者 A に公開可能であるかを確認する．

(3) 認証層は文書 X を要求する．

(4) 文書層は要求に基づいて必要なファイルを $get(R)$ によって分散層から取得し，それらを文書の形 $([X])$ に構成して認証層に渡す．

(5) 認証層は $\langle X, A\rangle$ にシステムの電子署名を付けて A に渡す．

文書 X の関連文書（仕様書や制御情報など）を得るには，$[X]$ 中のリンクを用いて，利用者が改めて閲覧を要求する．

5.2 文書の登録

利用者 A がファイル形式 X_{form} を持つファイル $\langle X_{text}\rangle$ に制御情報 D_{ctrl} を与えて POT に文書として登録しようとすると，以下のようなことが起こる．

(1) 認証層は A を認証する．

(2) 制御層は D_{ctrl} を制御文書として登録し，識別子 X_{ctrl} を得る．X_{ctrl} の制御文書や認証文書は，認証層が管理ポリシーに従って決定する．

(3) 文書層は文書 X_{form} が存在することを確認し，存在しなければ登録を拒否する．

(4) 文書層は，$create(\langle X_{text}\rangle, X_{form}, X_{ctrl}, A, n)$（但し n は $[X_{ctrl}]$ に記録された重要度）を承けて，文書 X を構成するファイルの作成を分散層に要求する．

(5) 分散層は適当なサイト（通常は登録を受け付けたサイト自身を含むであろう）の物理層にファイルを送付する．

(6) 文書層は上位の層に，今登録した文書の識別子 X を示す．

5.3 文書の自動的削除

保存年限が定められた文書 X を削除する場合に起きることは以下の通りである．

(1) 制御層のエージェントは，X の保存年限が経過したのを知ると，$owner(X)$ の資格で，文書層に対して $delete(X)$ を発行する．

(2) 文書層は，削除によってリンクが切れるなどの矛盾が生じないことを確認した上で，$delete(X)$ を実行する．

5.4 制御情報の変更: リンクの変更による方法

制御情報の変更には，親文書のリンクを変更する方法と制御文書に含まれる制御情報を変更する方法の 2 つが考えられる．リンクを変更する方法は A が文書 X だけの制御情報を更新するときに用い，制御文書に含まれる制御情報を変更する方法は，A が管理する文書 X を含む文書群の制御情報を更新するときに用いる．

まず，親文書のリンクを変更する場合に起こることについて説明する．

(1) 認証層は A を認証し，X_{cert} が A であることを確認する．

(2) 制御層は $\langle X'_{ctrl}\rangle$ を作成し，識別子 X'_{ctrl} を受け取る．

(3) 文書層の $updatelink(X, ctrl, X'_{ctrl})$ を実行する．

5.5 制御情報の更新: 制御情報を変更する方法

(1) 認証層は A を認証した後，X_{cert} が A であることを確認する．

(2) 制御層は，与えられた $\langle X'_{ctrl}\rangle$ の内容が $\langle (X_{ctrl})_{ctrl}\rangle$ の内容と矛盾しないことを確認した上で，要求された $setctrl(D, \langle D'_{ctrl}\rangle)$ を実行する．

5.6 利用者の登録

既にシステムに登録されている利用者 A が新規に B を利用者として登録するときに起こることは以下の通りである．

(1) 認証層は A を認証する．

(2) 認証層は，A から受け取った B に関する認証情報 $Cert$ で認証文書を作成する．この文書の廃棄期限は短期間としておく．

(3) B は POT にアクセスし，B の認証文書の保存期間を変更する．必要があれば，A に知られている B の鍵を，知られていないものに変更する．

なお，利用者は，システムにあらかじめ登録されている利用者からブートストラップする．

5.7 文書管理者の交替

文書 X の管理者 A が X の管理権限を B に譲渡するときに起こることは，以下の通りである．

(1) 認証層は A を認証し，A が文書 X の管理者で

あることを認証する．

(2) 認証層は，X の制御情報に，一定期間 B への譲渡要求が出ている」ことを記録し，B に権限が変更されようとしていることを通知する．

(3) B は POT にアクセスし，*updatelink*$(X, cert, B)$ を発行する．これは，A の譲渡要求の期間に限って許可される．

6. 実現可能性

本節では，第 4 節に示したアーキテクチャの実現可能性について検討する．

6.1 物理層の実現可能性

記憶媒体の物理的な寿命を超えて情報を永続させるには，適当な時間間隔で，古い媒体から新しい媒体に内容を移し，ファイル識別子と媒体上の所在の関係を変更する (**図 3**)．複写時に，複写先の媒体を新しい種類のものにすることによって，媒体の技術的進歩にも対応が可能になる．

また，複製，定期的なバックアップ，RAID などの技術もある．

単独の媒体の寿命を延ばす研究としては，HD-ROM[10] がある．HD-ROM は，媒体に物理的に微細な傷をつけることで情報を記憶するというもので，磁気を使用しないため，耐用年数が非常に長く取れるという．

6.2 分散層の実現可能性

永続的な命名体系を提供する枠組として現在提唱されているものには，URN[9] がある．現在はまだ仕様の詳細について議論が進められている段階であるが，永続的命名の標準になることが期待されている．

URN 以外の手段として，現時点で利用できるものに，PURL[12] がある．これは HTTP を利用したサービスで，特定のサイトに属する URL から実際の URL をリンクすることで実際の URL の移転の影響を吸収するというものである．将来的には URN ベースに移行することも可能だという．

完全に公開されているとは限らない文書をサイト間で交換する際に通信の安全性を確保するには，「分散層内用サイト間認証」が必要になる．また，分散層は，物理的には複数の写しがあるファイルの間の整合性を保証しなければならない．こうした要求を満たす道具立てとして，CORBA[4] を利用することが考えられる．

6.3 文書層の実現可能性

ファイルの構造，メタデータ，及びファイル間のリンク関係を表現する手段としては，XML[6] 及び XLink[8] がある（登録される文書の本文が XML でなければならないという意味ではない）．その理由は，

- XML はプレーンテキストの形で表現できるため，少なくとも記号レベルで読めなくなる心配がない
- XML を用いると，自由に文書形式を定義することができる
- XLink は，リンク元やリンク先の文書から独立したリンク情報を持たせることができる

などである．

更に，XML を用いてメタデータを記述する際の標準を定めようとする試みとして，RDF[11] や Warwick Framework[7] がある．

6.4 制御層の実現可能性

管理計画などの記述のフォーマットとしては，PICS[5] を援用できる．

制御情報に基づく動作の一部（文書の廃棄など）は，制御情報を読んで自動的に動作を行うエージェント（或いはデーモン）の存在を要求する．これはマルチプロセスの OS を用いれば容易に実現できる．

6.5 認証層の実現可能性

個人ごとの認証情報のフォーマットには，X.509[1] がある．また，システムにアクセスする者を認証する方法としては公開鍵暗号方式[2] がある．具体的な暗号方式については，技術革新に従って暗号方式を順次変更できる (3.3.3 節)．認証層の強度は暗号方式の強度に依存しているので，これは重要なポイントである．ただし，ファイル形式の仕様書と異なり，暗号処理系は POT システム自体が使うので，暗号方式の更新に伴って暗号モジュールを開発する手間は避けられない．

7. POT の利点

本論文で導入した POT 文書モデル及び POT アーキテクチャの利点を以下に挙げる．

電子文書の長期保管システムの一般的枠組 長期的に電子情報を管理するシステムの一般的な枠組を与えた．この枠組によって，個々の問題を整理して扱うことができる．

文書の永続性 普通文書のほか，永続性が要求される管理情報についても，永続性を保証することができる．

文書の遍在性 ネットワーク透過に文書を保存することが可能なので，特定の場所に依存することのない運用が可能になる．

検証可能性 各々の管理者の行為の履歴が残るので，将来，第三者が行為の妥当性の検証を行うことができる．

暗号技術への適応性 特定の暗号方式に依存していな

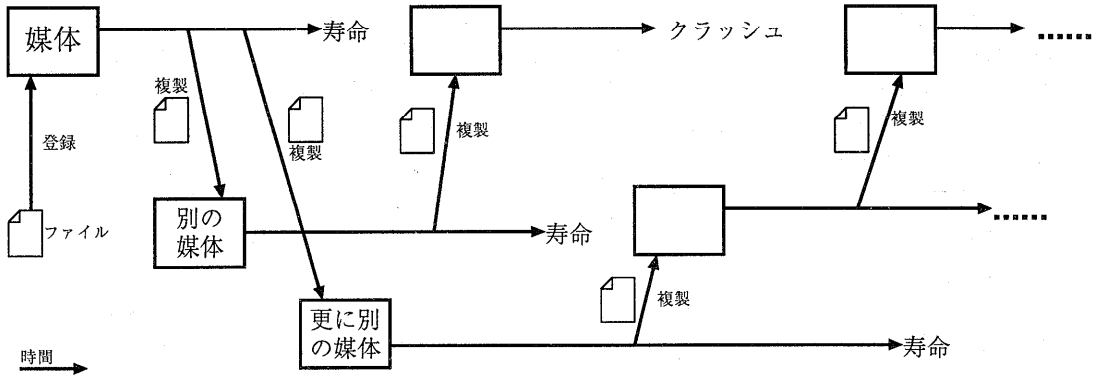


図 3 時間差を置いた媒体間の複写によるファイルの永続化
Fig. 3 Giving permanency to a file by copying among media with time lags.

いので，将来の暗号技術の発展に適応できる．

8. ま と め

本論文では，電子文書を長期的に保管するための枠組である POT について述べ，基本的な動作を見ることによってアーキテクチャ及び文書モデルの妥当性を検証した．更に，現存の技術による POT の実装にどれだけの現実性があるかについて考察し，要素技術が揃いつつあることを確認した．

今後，POT が様々な事態に正しく対応できる枠組であることを検証するために，本論文に基づいたプロトタイプの実装を行う予定である．また，システムのコストと文書管理の人的コストについても，定量的な分析を行う予定である．

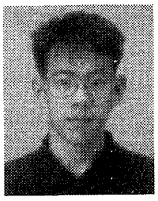
謝辞 著者に適宜有益な助言や指導を下さった，東京大学大学院総合文化研究科広域科学専攻 KTYY ゼミの皆様に感謝致します．また，電子文書管理の国際的動向に関して有益な講演をして下さった，David O. Stephens 氏に感謝致します．

参 考 文 献

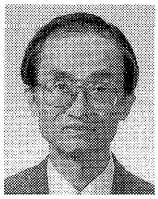
1) : InformationTechnology - Open System Interconnection - The Directory: Authentication Framework (1997). ITU-T Recommendation X.509 (1997) — ISO/IEC 9594-8:1997.
2) Diffie, W. and Hellman, M.: New Directions in Cryptography, *IEEE Transactions in cryptography*, Vol. 22, pp. 644–654 (1976).
3) Fielding, R. et al.: Hypertext Transfer Protocol – HTTP/1.1 (1997). RFC2068.
4) Group, O. M.: CORBA News, http://www.omg.org/news/index.html.
5) Group, W. P.I.: Platform for Internet Content Selection (PICS), http://www.w3.org/PICS/.
6) Group, W. X. W.: Extensible Markup Language (XML), http://www.w3.org/XML/.
7) Lagoze, C., Lynch, C. A. and Daniel Jr., R.: The Warwick Framework: A Container Architecture for Aggregating Sets of Metadata, http://cs-tr.cs.cornell.edu:80/Dienst/Repository/2.0/Body/ncstrl.cornell%2fTR96-1593/html/.
8) Maler, E. and DeRose, S.: XML Linking Language (XLink), http://www.w3.org/TR/1998/WD-xlink-19980303.
9) Moats, R.: URN Syntax (1997). RFC2141.
10) Stephens, D. O.: 文書管理における電子化の世界的動向 (1998). ARMA International 東京支部講演会.
11) Swick, R. et al.: Resource Description Framework, http://www.w3.org/RDF/.
12) Team, T. P.: Persistent URL Home Page, http://www.purl.org/.
13) Weibel, S. and Miller, E.: Dublin Core, http://purl.oclc.org/metadata/dublin_core/.

(平成 10 年 9 月 20 日受付)
(平成 10 年 12 月 27 日採録)

(担当編集委員 細野 公男)



伊戸川 暁 (学生会員)
1975 年生．1998 年東京大学教養学部基礎科学科第二卒業．現在，同大学院総合文化研究科広域科学専攻広域システム科学系修士課程在学中．



川合 慧（正会員）

1944年生．1967年東京大学理学部物理学科卒業．1970年同大学院理学系研究科物理学専攻修了．理学博士．1988年同大学教授．電子情報通信学会，ソフトウェア科学会各会員．

山口 和紀

1956年生．1978年東京大学理学部数学科卒業．1980年同大学院理学系研究科情報科学専攻修了．理学博士．1991年同大学助教授．ACM, IEEE CS各会員．