Vol.2014-DD-95 No.1
2014/10/3

アプリケーション仮想化環境における複合文書

大野　邦夫*,　　新　麗**

企業や官庁における情報システムの環境が急速にデータセンターに移行しつつある。その背景にはCRMやSFA等の大量マーケットデータを対象とするビッグデータの活用が挙げられるが、ネットワークセキュリティや情報漏洩への対処がオンプレミスのサイトでは困難になりつつあることも大きな要因である。さらにスマホの普及により、オフィスの情報環境と個人の情報環境の統合ニーズも顕在化している。そのようなニーズを背景にデータセンターにおける仮想化技術が進展しているが、利用者から隔絶された世界となっていることが問題である。本報告では、ゼロックスPARCで発明された複合文書技術に関し、OMGとW3Cによる標準化の経緯を通じて歴史的に考察する。さらにデータセンターにおける利用者情報をデータモデルに組み込んだアプリケーション仮想化環境に支援される複合文書サービスを想定し、厳格な管理と柔軟なサービスの両立の可能性を展望する。

Compound Document under Virtualized Application Environment

Kunio OHNO*,　　Ray ATARASHI**

Recently the environment of information systems in business and government area has moved to data centers. Though the background may be the big data application to the area of CRM, SFA, ERP, and etc., another important requirement is network security with malicious information attack and leakage, which cannot be protected within the existing on−premise sites. In addition to the existing office needs mentioned above, new requirements based on smartphones, which should be integrated to office systems are emerging. Though the virtualization technology in data center has been evolving based on those background, the world of data center which is isolated from the user is a problem. This paper describes the history of compound document technology, which was invented at Xerox PARC, and standardized by OMG and W3C. Then the possibility of flexible compound document service with strict information control will be expected through the user data model under virtualized application environment of data centers,

1. はじめに、

文書の電子化が行われるようになったのは、1960年代以降の半世紀である。テキストエディタからワープロへ、さらにDTPを経てHTMLによるマルチメディアWebへと進化したのであるが、文書の電子化の歴史的な意義を考える必要性を感じている。その理由は、最近のネットワーク監視ツールの進歩である。最先端の監視ツール提供企業は、従業員の人権を蹂躙しかねないツールを企業の情報システム部門に販売しているが[1][2]、社会制度が追いつかないうちにICT技術が飛躍的に進歩してしまったことが背景として挙げられる。

他方、米国政府は、議会の承認を経ることなく、世界中の個々人の通信履歴、通話履歴を秘密裏に収集していることをスノーデンが暴露した[3]。国立公文書館に象徴される米国の文書フィロソフィーとは相容れないものと感じるが、米国社会の理想と現実の乖離は、今日の電子化文書、デジタルドキュメントに問われている問題そのものでもあろう。その鍵を握るのは、今や個々人の生活履歴を包含するビッグデータを管理するデータセンターであると感じている。

* 職業能力開発総合大学校
Polytechnic University

**IIJイノベーションインスティテュート
IIJ Innovation Institute

本報告では、データセンターにおける最新の情報管理としてのアプリケーション仮想化環境を想定した複合文書について、歴史的な経緯を踏まえた上でその可能性について検討を試みる。

2. 複合文書に至る歴史

2.1 文字・図形・画像編集の端緒

電子化文書の概念は幅広い。歴史的な経緯を考えると、1960年代にメインフレームコンピュータによる電算写植による組版システム（CTS）が開発され、大手の印刷出版企業に導入されたことが身近な生活に関わる発端であろうと思われる。文字・図形・画像をコンテンツとしてGUIでページレイアウトすることが可能なレイアウトシステムを実現したのは、ゼロックスPARCのALTOパーソナル・コンピュータであった[4]。

1970年代に開発されたALTO上のBravoエディタは、画面上の表示内容がそのまま印刷ページとして得られるWYSIWYG（What You See Is What You Get）を実現し、その技術に基づき開発された商品がStarワークステーションであった[5]。Starのコンセプトは斬新でそのGUIの設計思想は、後の各種ワークステーションやPCのヒューマンインタフェースの基本になった。

Starが出現した1980年代前半は、インテルの8086やモトローラの68000に代表される16ビットのマイクロプロセッサが導入されて、PCや通信端末の高機能化、高性能

化が進展する一方、日本では第五世代コンピュータプロジェクトが立ち上がり、人工知能ワークステーションの実現を目指した研究が行われた時期であった。その当時著者の一人（大野）が所属したNTTの電気通信研究所宅内器機研究部では基礎研究部で試作完成したリスプマシンELISの商品化を目指した開発が進められていた。

2.2 電子化文書の本質

ELISの実用化を推進していた1985年にNTT本社から緊急案件としてゼロックスPARCとの共同研究の打診があった。日米貿易摩擦の余波である。テーマは、ドキュメントの基礎（Foundation of Document）という将に基礎的なものであったが、取りあえずPARCの担当者と折衝することになった。PARCの担当者は、デイビッド・レビー氏で哲学者的な物静かなユダヤ人研究者であった。資料を読み説明を聞いた印象では、電子化文書は、単にWYSIWYGで印刷文書が得られるようなものではなく、コンピュータのデータ構造を二次元画面にマッピングする演算操作が本質であるというものであった。単純化して紹介すると、

$$X = F(Y)$$

というベクトル・マトリクス方程式を想定し、ベクトルXがレイアウトの属性と値のペア、Yがコンピュータ上の文書のデータ構造、Fが変換操作ということである。この研究の成果により、文書構造をレイアウト構造に変換する汎用的なアルゴリズムを導出する可能性を理解した。ELISへの流用を考え、成果物としてそのアルゴリズム記述をLISP言語で提供する契約書としてまとめた。

個人的に興味を持ったのは、複雑な文書編集操作を数学的なマトリックス演算として扱うセンスであった。その操作は象徴的にホムンクルス（Homunculus）と表現されていたが日本には無いPARCらしい斬新な文化を感じた。

その後、ELISの事業化のために新規事業開発室に異動になり、後任に引き継いだが、後任者が契約内容を大幅に変えてしまった。後にレビー氏から、その話を聞かされたが文書や契約に対する日米の文化の相違を考えさせられた。一度結んだ契約内容を大幅に変えるというのは米国ではあり得ないことなので（そのために契約を結ぶ）、PARC内部での説明と事務手続きが大変だったとのことであった。

2.3 インターリーフDTPのインパクト

ELISのビジネスは、NTTインテリジェントテクノロジ（NTT−IT）社で進められたが、通研の支援が無かったこともあり、ELIS上での文書システムの開発は行われなかった。他方、ライバルのSymbolicsは、GeneraというLISP環境に”Document Examiner”や”Concordia”という文書管理システムを構築し、実用性に優れたマニュアル参照システムを提供していた。

NTT新規事業開発室では、NTTグループ企業のドキュメント制作や管理を外注で引き受ける事業会社を企画していた。そのプロジェクト経由で、米国のインターリーフ社がツールを日本語化して日本で事業を行うべくパートナーを探しているという情報を入手した。新規事業開発室としては、その事業可能性を調査したいとのことで協力を求められた。インターリーフ社のDTPツールについては、マニュアル制作関係者からその卓越した優秀性を聞いていたので、NTT−IT社の事業拡大の観点から調査に協力した。1989年の秋にボストンのインターリーフ社を訪問し、エンジニアやマーケッティングのSEとディスカッションしたが、興味深いことにコアの開発はLISPで行われていた。

その翌年の4月にNTTヒューマンインタフェース研究所に異動になり、NTTグループ事業推進本部（新規事業開発室の後継組織）と連携してインターリーフの日本語化と日本国内の市場調査のプロジェクトを立ち上げることになった。その過程でインターリーフ社のDTPシステムを始めとするツール類について詳細を知り得たが、かつてELIS上で開発を意図した仕様をはるかに上回る完成された機能を実現しており、日本語化できた暁には幅広い市場に対してビジネスが展開できそうな魅力を感じた。1990年末に、インターリーフ社とNTTは事業化を目指して検討チームが設立された。

2.4 Interleaf5のアクティブドキュメント機能

その後は、毎月検討チームのミーティングが日米交互で開催され、隔月に米国に出張する多忙な日々を過ごしたが、Interleaf Lispでカスタマイズする豊富な機能について知る機会を得た。特に感心したのはTCP/IPによるRPCとNFSを活用するネットワーク機能であった。

Interleaf社のDTPシステムのTPS4は、CADやコンピュータ・グラフィックスのアプリケーションのデータをフィルタ経由で自分の図形データ形式に変換して取り込むことが可能であった。ファイル形式を変換するフィルタツールはInterleaf Lispで開発された。さらにNFSを用いることにより、遠隔のコンピュータ上のファイルにアクセスしてDTPシステムに取り込むことが可能であった。この機能を用いてボーイング社やキャタピラ社は、設計図面を含む技術文書をインターリーフで管理していた。

TPS4の後継システムであるInterleaf5は、このフィルタツールの機能をさらに汎用的に拡張した。この機能はアクティブ・リンク機能と呼ばれ、その後のOMGやW3Cが目指したコンポーネントウエアとしての複合文書機能そのものであった。アクティブ・リンク機能は、DTP上のフレーム画面を通じて遠隔アプリケーションを起動してその結果をフィルタ経由で取り込むクリエイティブ・リンク機能、遠隔アプリケーションの最新データにアクセスしてそれを取り込むライブ・リンク機能、状況に依存してデータを選択的に取り込むことが可能なインテリジェント・リンク機能に大別され、NFSで管理可能なコンピュータネットワークシステムの多様なデータを一元的・系統的に管理することを可能としていた。

さらにインテリジェント・リンク機能の選択機能に、エキスパートシステム的なルールベースの推論機能や決定木による選択機能ををを組み込んだDTP応用システムも開発された。そのような機能はレイヤード・アプリケーションと呼ばれた。そのような経緯からInterleaf5はアクティブ・ドキュメントと称された[6]。

アクティブ・ドキュメントにおけるアクティブ・リンク機能は、Unixワークステーションだけでなく、NSFをサポートするPCやメインフレームのアプリケーションにも対応可能で、スプレッドシートやRDBアプリケーションにも適用可能であった。

アクティブ・ドキュメント機能を有効に使用した事例としては、先に述べたボーイング社の航空機設計部門が挙げられる。そのシステム構成を図1に示す[7]。航空機の設計図面は3次元CADのCATIAで管理され、関連データは

CATSTEPデータベースで管理されている。機体材料のデータはMATDBで、部品の設計データはFASDBで管理され、機体の強度計算やシミュレーションはSHRBEAM、IAS、SAといったサブシステムで処理される。これらのデータはInterleaf5のアクティブ・ドキュメント機能により管理され、技術ドキュメント、設計マニュアル、ユーザーズガイドなどのドキュメント作成、配信、管理に使用されていた。

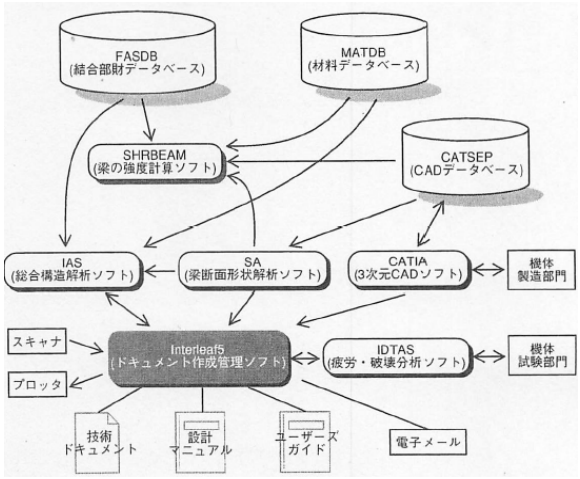

図1　ボーイング社における**Interleaf5**活用事例

3. 複合文書の標準化

3.1 OMGのCORBA

オブジェクト指向プログラミン技法は1960年代のSIMULAに端を発し、Smalltalkを通じてLISPやC、Pascalなどのプログラミング言語に浸透した。その後、1990年代に入って、ネットワークOSにオブジェクト指向技術が適用された。この技術は「分散オブジェクト」と呼ばれ、OMG（Object Management Group）のCORBA（Common Object Request Broker Architecture）として仕様化された。

オブジェクト指向プログラミングでは、先ず実行対象の集合概念としてのクラスを定義し、その集合の変数と処理手続きとしてのメソッドを定義し、具体的なオブジェクト（インスタンス）を生成し、そのオブジェクトにメッセージを送信（メソッドの実行）して結果を得るという一連の処理となる。分散オブジェクトは、上記の「具体的なオブジェクトにメッセージを送信して結果を得る」という操作プロセスにクライアント・サーバ・システムを対応付けてモデル化したものである。クライアント・サーバ・モデルは、遠隔のサーバーに対して、リクエストを送信し、結果の値を得る操作だからである。このモデルは下記のように表現され操作シグニチャと称される[8]。

$$\omega：（\chi_1:\sigma_1, \chi_2:\sigma_2, ・・・\chi_n:\sigma_n）\\ \rightarrow（y_1:\rho_1, y_2:\rho_2, ・・・y_m:\rho_m）$$

ただし、ωは操作の名称、χは型σのリクエストパラメータ、yは型ρの結果のパラメータである。パラメータ数nは1以上、mは0以上である。なおこの操作は、

$$Y = \Omega(X)$$

という代数演算に対応付けることが可能であり、PARCのレビー氏におけるX＝F(Y)の演算のOMGによる汎用オブジェクトモデル表現と考えることもできる。なおここで重要なのはパラメータの型である。入力パラメータの型Σが結果のパラメータの型Pに対応付けられて変換される。この視点に立つと、SGMLのDTDによる文書構造の型を、代数的な演算のパラメータの型に対応付け、レイアウト表現の型（後のDSSSL）に変換するモデルへの理論付けの思想が生まれるのである。

3.2 演算操作と型の普遍性

CORBA仕様における数学的な原理である操作シグニチャのモデルは、オブジェクト指向のメッセージ・パッシング乃至はクライアント・サーバモデルのリクエスト・レスポンスという概念に対応する。だがそれよりも、遠隔手続き呼出し（RPC）の要求と結果と見なす方が現実の機能に即している。さらに遠隔手続き呼出しの概念は、ローカルな手続き呼出し、すなわち一般的なサブルーチン・コールの概念も包含する。

ということは、計算機プログラムの実体は型を伴う代数学的な演算の集合ということになる。この概念に最も適合するのは関数型言語であろう。その観点からするとLISPは最も適合しそうであるが、一般には型付けされていない（強いて挙げるとアトムとリスト）。Common Lispではコンパイラ効率を上げるために既存の言語の型付けをベースに系統的に型を追加した。LISPの言語仕様を型に基づいて見直し、CLOSをベースに汎用オブジェクト指向システムとして体系化した点にCommon Lispの真価がある。この仕様の完成度は非常に高い。拡張可能な履歴書システムの実装[9]をはじめとする職業大でのプログラム作成に、Common Lispを用いたが、30年近く前に制定された詳細な言語仕様が最新のAllegroCommonLispの環境で完璧に動作し、仕様書自体が非常に有効であった。コンピュータ分野でこのような事例は希有であろう。

型推論を含む強く型付けされた関数型言語としてはML（Meta Language）[10]があり、Standard ML of New Jerseyの処理系が入手可能である[11]。モデルとして文書型を考察するためには興味深い仕様であるが、実用システムには無理がある。実用的な型付け言語となるとやはり手続き型言語のC、C++、Objective−C、Javaである。計算機環境は、実用的には環境依存の状態パラメータが必要であり、理想的な関数型言語によるプログラミングは実用的にはどうしても無理がある。

3.3 OMGの複合文書

OMGは図2に示す標準化のためのOMA（Object Management Architecture）参照モデルを定めていた[8]。このモデルは、4部分から構成される。オブジェクト・リクエスト・ブローカ（ORB）は通信メカニズムを提供するが、ORBの実装仕様がCORBAである。オブジェクトサービス（OS）は、ORBが提供する基本サービスで、CORBAオブジェクト（図の丸印要素）および環境依存の擬オブジェクト（図の矩形要素）から構成される。ファシリティは共通ファシリティ（CF：CommonFacitity）、またはCORBAファシリティと呼ばれ、サポートすることを義

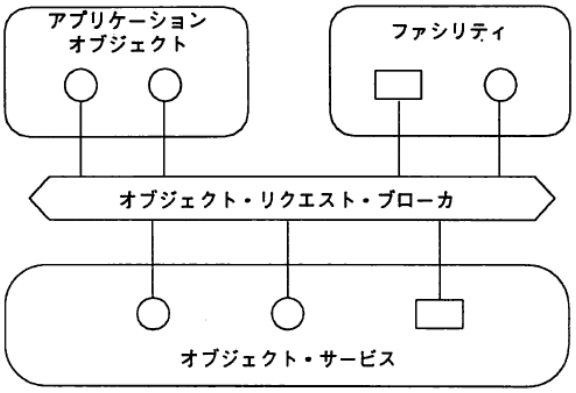

図2　OMA参照モデル

務づけられてはいない共通的なサービスである。GUIや文書環境、ワークフロー、カレンダー、情報管理、システム管理などが想定された。アプリケーション・オブジェクト（AO）は、各種の技術分野や業界毎のサービスで、当初は標準化の対象外とされたが、後に業界毎のAPIを決めるようになった。

複合文書は、ファシリティの要の機能であり、GUIと文書作成、編集、管理を含む。具体的な複合文書は、アップルのOpenDOCの仕様が共通ファシリティ・タスクフォース（CFTF：Common Facility Task Force）で審議され、技術委員会（TC：Technical Committee）で承認された。

OMGの複合文書は、コンテナ（ウインドウ）とパーツ（コンポーネント）から構成され、パーツは文字入力、スプレッドシート、グラフィックツール、映像、アニメーションが用意されていた。パーツツールとしては、作成編集ツールとしてのエディタ、表示印刷用のビューワ、サーバ機能を提供するサービス、エディタとサーバを統合したコンテナアプリケーションが実装環境として用意されていた[12]。要するに複合文書は構造とレイアウトを関係付けると共に構造を構成するパーツを動的なまま管理し、スナップショットを静的なドキュメントとして参照可能とするシステムである。

OMGの複合文書は、OMG内では審議が進み標準仕様となったのであるが、市場はマイクロソフトのOLE（Object Linking and Embedding）がアップルのOpenDOCに比べると圧倒的に優勢であった。アップルのOpenDOCは、マイクロソフトのOLE（Object Linking and Embedding）との競争に対抗できず消滅したが、同時にOMGの複合文書も普及の可能性が無くなった。

3.4 W3Cのドキュメント・オブジェクト・モデル

OMGの複合文書は、実際に普及したと想定しても、既に業界で標準化されていたSGMLとの整合が問題になったと思われる。他方、SGMLの世界は、その実装DTDとしてのHTMLを通じてインターネット上の文書標準の枠組みとして定着し、さらにWebを構成するデータ要素としてリファインされたXMLとなって、文書・データ用の標準フォーマットとして普及・定着した。XML化に伴い終了タグの付与がSGMLに比較した最も目立った変化であるが、さらに変数域を定義する名前空間、要素へのアクセスAPIとしてのDOM、データ型を定めたXML Schemaなどが仕様化された。

興味深いことに、DOMは言語中立を意図してCORBAのインターフェース定義言語（IDL: Interface Definition Language）で記述された。要するに、DOMのAPIは、CORBAオブジェクトの操作シグニチャに準拠しているのである。しかし、型はIDL型ではなくXMLSchemaで定められた型となるので、厳密にはCORBAの操作シグニチャではない。

XML Schemaのデータ型をIDL型に合わせておけば、興味深い発展ができたのではないかと思う。この趣旨は、CORBAが普及してHTTPでなくIIOP（Internet Inter ORB Protocol）が使われる可能性を追求するというよりは、XMLを活用するアプリケーションの枠組みをプログラム言語独立に設計し、各種言語に実装する可能性、すなわちアプリケーション仮想化の観点からである。

この可能性をDOMワーキンググループ議長のローレン・ウッド女史に直接質問したが「それは全く考慮していない」と一蹴された。議論を続けたかったのであるが彼女はそれを避けた。彼女はCORBAを知らなかったのかもしれない。同じ議論をジェームス・クラーク氏にぶつけたら、彼は「XML Schemaの型は冗長なので、冗長性を排し論理的に整合されたCORBAのIDL型適用は興味深いアプローチだ」と語り、XML Schemaの用法を数学的に妥当にし得るという見解を述べてくれた。

3.5 W3CのCDFとジャストシステムのxfy

2004年にW3CがXMLによる複合文書関連の標準化を目指してCDF（Compound Document Format）ワーキンググループを立ち上げたが、その経緯の詳細についてはDD研で報告した[13]。

ジャストシステムのxfyは、一太郎のコンポーネントをXML化しJavaで実装することを試みた一太郎ARKに端を発する。このことから分かるとおり、xfyはオフィス情報システムにおけるXMLデータやコンテンツを、オフィス文書として統合することを可能にする革新的な製品であった。その普及戦略として、CDF WGに提案して国際標準とすることが当初の狙いであった。

しかし、CDF WGの狙いはオフィス文書ではなく携帯電話画面にあった。NTTドコモのiモードに刺激された欧州の携帯電話関係企業が、インターネット・コンテンツの世界に携帯電話を融合させるべく携帯コンテンツの標準化に着手したのである。それを察知してxfyの携帯版の開発をジャストシステムの幹部に提案した。だが当時の携帯電話のアプリケーション環境は30MB程度の容量しか許されずXMLパーサーを搭載するのに苦労していた状況であった。従って1GB必要とされたxfyにその可能性は所詮乏しかった。

結局、xfyはオフィス文書向けのXMLコンテンツを統合する複合文書システムとして開発が進められた。金融業界や証券業界向けにXBRL（eXtensible Business Reporting Language）に特化された製品（xfy for XBRL）が開発され、EDINET向けの有価証券報告書作成用パッケージがサンプル出荷された。さらにXMLデータベースでコンテンツ管理するクライアント・サーバ版のxfy for XBRLが銀行向けに開発されたが製品化には至らなかった。しかし、

XMLコンテンツの複合文書システムとしては他に例を見ない興味深い機能を実現した製品であった。xfyの最大の特徴は、XMLの世界でWYSIWYGを可能とした点にある。このメカニズムは、XSLTの変換機能を双方向化したXVCD（XML Vocabulary Connection Descriptor）により実現されている。

XSLTはXMLの論理構造を別の構造に変換する処理を行うが、それは一方向だけの変換である。XHTMLが、SVGやXFormsのようなXMLのデータを包含する場合、Webブラウザ上にはSVGに対応する図やXFormsに対応する表が表示される。そのためにWebブラウザは、SVGやXFormsを表示するビューワを内蔵して表示する。ビューワの代わりにXMLデータを編集可能なSVGエディタやXFormsエディタを持たせ、XHTMLが包含するコンポーネントとしてのXMLデータを入力可能とし、さらにXHTMLのWeb画面として表示可能としたものがXVCDである。従ってXVCDの基本アルゴリズムはXHTMLのWebレイアウトに構造変換するXSLTと同様である。XHTMLのWeb向けレイアウト構造をX、元のXMLコンポーネントを含む論理構造をYとすると、XSLTの変換は、

$$X=F(Y)$$

と表記可能である。この表現は、2.2節で述べたPARCのレビー氏のコンセプトと全く同様であり、その具体的な実現と考えて良いであろう。さらにX=F(Y)というコンセプトは、xfyの名称の由来でもある。

3.6 XBRL処理系としてのxfy

xfyをローカルにではなくクライアント・サーバ系のクライアントとして用いることも検討された。その構成例を図3に示す[14]。

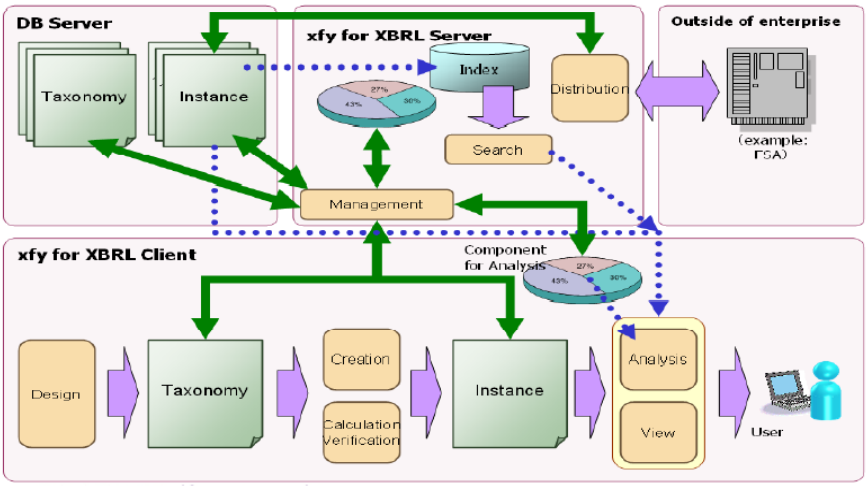

図3　クライアント・サーバ系向けのxfy

クライアントのxfyは、EDINET用の場合のXMLデータを遠隔のサーバからダウンロードし、処理が完了したらアップロードするような粗結合のシステムである。というのは、xfyがCDI（Compound Document Format for Inclusion）形式の仕様であり、携帯電話画面用に正式勧告化されたCDR（Compound Document Format for Reference）のように参照が基本にはなっていないためである。この制約は、XVCDがローカルでの処理に基づいていることに起因する。

図4は、クライアント・サーバ用のxfyを用いて銀行の融資業務支援システムを構築する場合の構成例を示す。操作者による入力編集作業が要求される箇所毎にクライアントとしてのxfyが使用されている。

3.7 複合文書の総括と反省

以上、PARCの思想、インターリーフのDTP、OMGのOpenDOC、W3CのCDFとジャストシステムのxfyについて紹介した。複合文書の定義は必ずしも明確ではないが、文字、図形、画像などの文書要素を動的なコンポーネントとして管理する機能を持つ電子化文書の枠組みと考えれば良いであろう。さらに文書の本質として証拠や契約としての意味も必要なので、静的な文書への変換機能も必要である。要するに成作編集用のエディタと清書参照用のビューワに対応するフォーマットを必要とする。ワープロやDTPの時代には印刷文書がビューワの役割を担ったが、現在はPDFが清書フォーマットの役割を担っていると言える。複合文書のデータ構造について簡単にまとめると表1のようになる。

表1　複合文書のデータ構造

論理構造	レイアウト構造	清書構造
Interleaf Lisp	Interleaf Lisp	Printerleaf
SGML	DSSSL	SPDL（EPS）
XML(CDF)	XSL・CSS	PDF
XML(xfy)	XVCD・CSS	PDF
XML(HTML5)	XSLT・CSS3	PDF（EPUB3）

論理構造とレイアウト構造の変換は、数学的な概念が興味深いが文書の型概念を明確化する必要があり、オブジェクト指向プログラミングにおけるクラスのように単純ではない[15]。

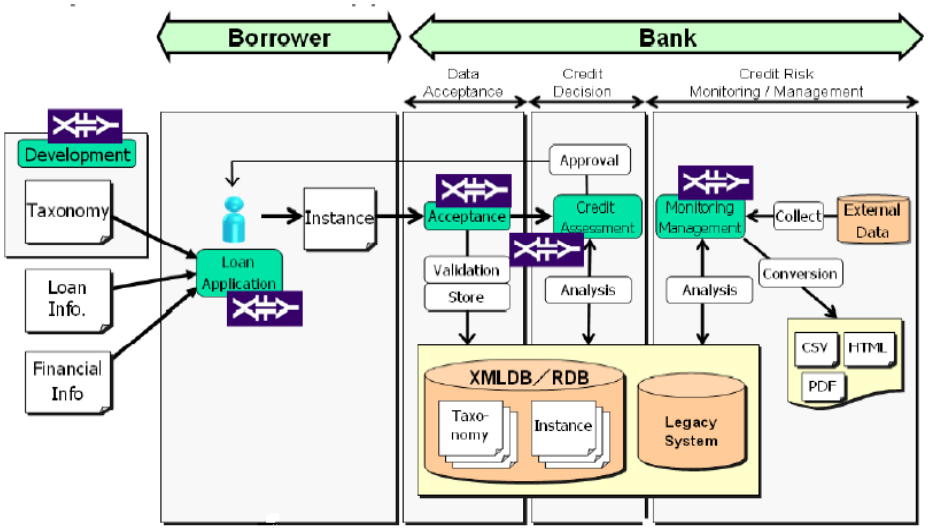

図4　xfyによる銀行の融資業務支援システム構成例

LISPは柔軟な計算アルゴリズムに基づく関数型言語なので、環境依存のデータも含めその型変換の処理を比較的容易に実現する。InterleafとDSSSLはLISPの機能を有効に活用して変換を実現した。SGMLのDSSSLに相当するXMLの世界におけるXSLは、構造変換のXSLTとフォーマット処理のXSL−FOに仕様を分けたが、XSL−FOは処理が煩雑でアンテナハウスの有償の処理系を除き実装製品は普及していない。その代わりにCSSの機能が拡大してきた。

HTML5が複合文書かと言われると異論があるかもしれないが、文字、図形、画像、映像、音声を包含するブラウザなので複合文書の一種には違いない。しかもCDFによる標準化成果も盛り込んでいる。とは言え、従来のHTML4.2とXHTMLの両者を継承・統合し、映像、音声といった動的なメディアまで包含したので極めて複雑な仕様になっている。HTML5の技術を電子書籍向けのビューワに改定したEPUB3もその範疇であろう。ただしWebブラウザの場合清書フォーマットとの関係が不明確である。CSSにおいては静的なページレイアウト概念が明確でなく清書構造の定義があいまいなためである。

レイアウト構造から清書構造への変換は、プリンターのドライバーのような処理であり、紙に印刷する代わりにイメージ画像を作成するシステムである。Interleaf社は、Printerleafというフォーマットを持ち、WorldViewというビューワを提供していた。Interleafに特化されていたので、PDFに比べると遙かにコンパクトであるがが画像品質は優れていた。SGMLの世界におけるSPDLの標準化は、アドビのPostScriptの普及でEPS（Encapsulated PostScript）が事実上の標準となった。

コンポーネントウエアとしての複合文書の特色は、コンポーネントへのアクセスメカニズムである。複合文書のネットワークアクセス機能について簡単にまとめると表2のようになる。

上記において安定な仕様はInterleafのNFSだけであることに注目したい。OMGのCORBAオブジェクトは普及しなかった。MicrosoftのOLEは、Windowsの世界に閉じた使い方であった。DOMは、Webの進展と共に膨大な仕様が追加され安定な仕様とは言い難い。複合文書が普及しなかった原因もこのあたりにありそうだ。

表2　複合文書のネットワークアクセス機能

枠組み	コンポーネント	アクセス方法
Interleaf	ファイル	NFS+フィルタ
OMG	CORBAオブジェクト	CORBAリクエスト
Microsoft	DCOMオブジェクト	OLE
W3C CDF	XMLデータ	DOM

4. アプリケーション仮想化と複合文書

4.1 シスコのACI

アプリケーションの仮想化は、シスコが系列製品のACI（Application Centric Infrasteucture）の開発や運用に関連して提唱している思想であるが、先に述べてきた複合文書の問題を解決し、新たなサービスの方向性と可能性を感じさせる[16]。ACIの構成を図5に示す。

シスコのACIは基本的に、ファブリック、APIC（ApplicationPolicyInfrastrctureControl）、プロファイルの3種類のモジュールから構成されている。ファブリックはコンピュータラックやブレードを相互接続したネットワーク実装モジュールである。プロファイルは、正式には、Application Network Profileであるが、仮想化されたアプリケーションのモデルである。このモデルは3層クライアントサーバとしてのモデル例である。

APICは、プロファイルに基づくポリシーを、ファブリックに適用するポリシー制御システムで、SDNにおけるOpenFlowアーキテクチャのNorthbound APIをサポートし公開している。ACIはさらにSouthbound APIも提供し、サードパーティのネットワーク・サービス・ベンダーが提供するデバイスのためのポリシー制御をAPICを介して実装することが可能である。

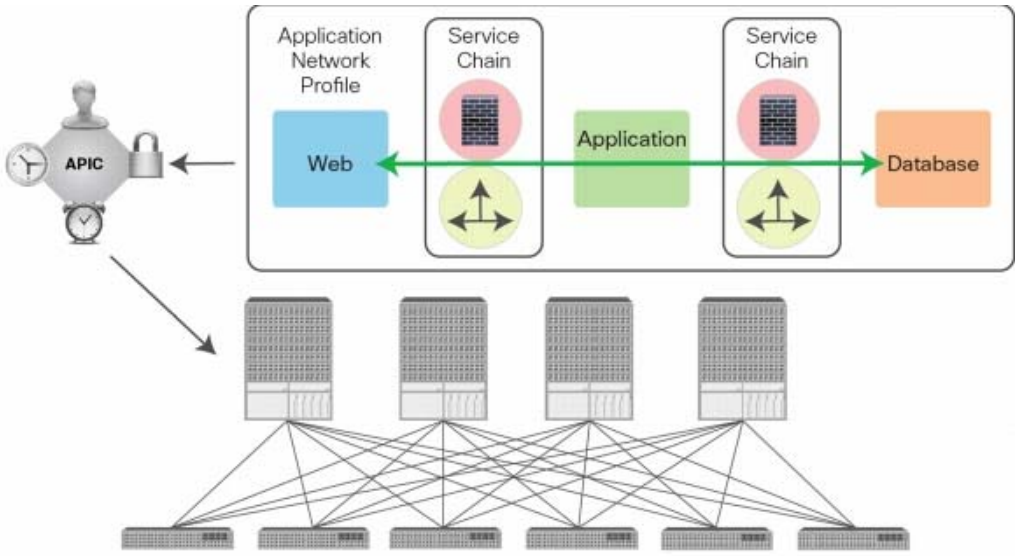

図5　シスコのACIにおけるアプリケーション仮想化の機構

4.2 アプリケーション仮想化のメカニズム

APICとプロファイル、Northbound APIとの関係の詳細は必ずしも明確ではないが、既存のアプリケーションサービスにおける通信量や処理量といった情報管理の量的な制御にNorthboundAPIが関与するのに対し、プロファイルは、アプリケーションのサービス内容やデータ内容の抽象化を通じたカテゴリに関する情報をAPICに提供すると考えられる。なお詳細は不明なので、この機能を活用する複合文書の可能性など、検討の対象というよりは見当を付ける程度の議論であることをお断りしておく。

アプリケーション中心のポリシーについては、シスコの白書によると下記のように書かれている[17]。

(1)ACIポリシーは、トップダウンの命令的な管理ではなく、自律分散的な推論機能を活用する束縛理論に基づくオブジェクトモデルで管理される。

(2)サービス対象の組織やグループをテナントという概念で区分し、その配下にニーズに応じたコンテキストを構築し、コンテキストに準拠するアプリケーションカテゴリを整備する。

(3)オブジェクトは、アプリケーションカテゴリを活用する実体で、各種APIに対応したエンドポイント（EP）、エンドポイント グループ（EPG）、およびその関係を定義するポリシーから構成される。

EPは現状では、NICや仮想NIC、IPアドレス、DNSなどのレベルであるが、将来的にはプログラムのメソッドに対応させるとのことである。そうなると言語独立なCORBAのインタフェース定義かXMLのOWLのようなオントロジ定義とする必要があるだろう。プロファイルは、アプリケーションカテゴリを定義・支援するシステムとして位置付けられている。

4.3 今後の複合文書への期待

アプリケーション指向ということは、アプリケーション・ニーズに応えることが可能な機能を提供することに他ならない。かつて分散オブジェクトの世界でアプリケーション・ニーズに応える方法論としてCORBAを活用する複合文書がOpenDOCを通じて検討された。さらにWebの世界でXHTMLとXMLコンポーネントを用いてアプリケーション・ニーズに応える複合文書の方法論としてCDFが検討された。ジャストシステムのxfyもその事例であった。今後のアプリケーション仮想化の方向性もニーズに基づき予想することが可能である。

今後の複合文書に関しては、下記のようなニーズが想定される。

(1)オフィスでは今後もA4サイズの静的な文書情報が必要とされるであろう。

(2)A4サイズの静的な文書情報の作成ツールとしては、MSのワードが主流であるが、今後はHTML5のWeb情報をA4サイズにレイアウトするニーズが増大する。

(3) クライアントをWebとし、ビジネスロジックとデータモデルで3層化するアプリケーションの仮想化モデルが進展する。

以上のように考える背景は、図3と図4で示したxfyの経験に基づいている。シスコが当面の仮想化モデルとして提案した3層システムは現状の多くのサービスには妥当と感じる。さらにアプリケーションの仮想化にとって、今後重要になるのは下記のようなデータモデルであろう。

(1) データモデルとして、企業や公的機関などのビジネスデータ、統計データがサービス付加価値のために活用される。

(2) その一環として人的組織情報が活用され、プライバシ、個人情報保護が極めて重要なニーズになる。

(3) アクセス権を考慮した情報の配信と管理にとって複合文書は有効な媒体である。

文書管理はワークフローを伴うので責任者が明確化された上で情報が配信される。特に個人情報を重視すべきサービスは生のDBによる情報よりは複合文書で管理する方が

社会的に馴染むと思われる。この特徴は組織を人間的に管理する上で大きなメリットとなる。すなわち、

(1) 組織構成員の各種データ（DBや各種文書）へのアクセス権を職場の運営規則や習慣に基づき分類管理する。

(2) アクセス権は、組織としての権限だけでなく、個人のプロファイル情報（スキル、資格、趣味、特技など）も活用する可能性を検討する。

(3) 組織構成員が組織として発信する情報（メール、Webアクセス、電話発信）に対しては、管理責任者を明確化し、履歴を管理すると共に、適宜フィードバックを行う。

上記の3項目については、職場としての情報管理・運営に関するルールを確立する必要がある。最近Motex社のLanscope Cat[1]とSKY−SEA[2]のネットワーク管理ツールのデモを参照し、上記の問題を痛感した。アプリケーション仮想化を企画・推進する立場の関係者は職場としての情報管理・運営ルールの雛形を提供することが望まれる。

5. 考察

以上、複合文書の歴史、データセンターの状況を背景に文書の電子化に関する概念の展開や動向を述べたが、PARCのALTO以来の40年の歴史の中で複合文書は人間や組織に深く関わっており、一般生活者にとってのコンピュータとネットワークの歴史そのものと言っても過言ではない。複合文書の構造とレイアウト演算やそれに伴う型の問題はHTML5やCSSの動向を鑑みるとあまり進展していない。アプリケーションの仮想化を想定するとこの課題に取り組む文書モデルの構築が期待される。

データセンターを背景とした今後の複合文書の役割としては、地域コミュニティの生活支援なども視野に置くべきであろう。雇用や就業のための求人データや人材データ、医療や介護のための電子カルテ、幼児や学童の保育や教育のための情報、高齢者・障害者を対象とする要介護や要支援情報などのためのアプリケーション仮想化も検討されるべきであろう。

特に今後重要と思われるのは、機密情報に対する扱いである。機密情報は、本来公開すべき情報であっても、その一部に公開が懸念される情報があると、全ての情報が機密になってしまい、熱力学第二法則におけるエントロピーの如く機密が増殖する。これを防止するためには、アクセス権を厳密に管理してその複合文書の属性として可能な限り公開して管理することが望まれる。

組織毎の機密情報の管理の仕方は多様であり、それを組織構成者の資格やスキルに対応付け、アクセス権を設定することが、今後のビジネスや、行政では重要になる。特にグローバル化で、外国人労働者を受け容れたり、外国企業との契約などにおいて、アクセス権管理は重要である。その機能の実現に、データセンターは有効な機能を提供すると思われるが、その実現のために複合文書は生のDBによる監視ツールよりは遙かに有効であろう。

6. おわりに

本報告の視点から、今回の研究会のセッションテーマである「チーム活動を支援するドキュメントコミュニケーション」について考えると、チームを構成するメンバーの人的情報を支援する枠組みの提供が挙げられる。チームと言えども、責任者が明確化され、役割が分担され、個々人の最適な活動を支援することが要請されると思うからである。

特に今後の地域コミュニティやNPOのような緩やかな組織においては、強固な組織とは異なる自由な個人の自発性を活用・支援するチーム活動が重要になる。そのような場面に、アプリケーション仮想化の枠組みによる人的属性情報を活用する複合文書の適用が有効と思われる。ただし、そのためにはチームメンバーのプロファイルの登録やその情報の活用の許諾に関する事前の取り決めが必要となる。そのようなサービスに際してはデータモデルに融通性を持たせるコンシェルジュのような支援機能が期待される[18]。

さらに、そのような個別的・具体的な取り決めのためには仲介するコーディネータが必要であり、その人材の資質・経験や対人スキルが重要になる。別の見方をするとチームメイトから見たコーディネータの人格そのものであろう。今後の日本社会において、コミュニティにおけるそのような人材の育成が大きな課題である。

文献

[1] http://www.motex.co.jp/products/cat/

[2] http://www.skyseaclientview.net/

[3] グレン・グリーンウォルド; “暴露:スノーデンが私に託したファイル”, 新潮社（2014.5）

[4] 喜多千草; “起源のインターネット”, 青土社（2005）

[5] D.C.Smith et. al.; “Designing the Star User Interface ”, BYTE,Vol.7, No.4, pp.242.282（April 1982）

[6] 大野邦夫; “アクティブ・ドキュメントにおけるLISPの活用”, Forum Proceedings on JapanPractical Application of Lisp Forum & Exhibition’91（JPAL’91）,（1991）

[7] 大石進; “オーバービューオブInterleaf5−Part2”, SuperASCII, Vol.3, No.11（1992.11）

[8] リチャード・ソーリー（大野他訳）; “オブジェクト・マネージメント・アーキテクチャ・ガイド − OMAG2.0”, 創研プランニング（1994.1）

[9] 大野邦夫, 角山正樹；”拡張可能な履歴書管理システムの実装に関する検討”, 職業能力開発総合大学校紀要（2011）

[10] Jeffery D. Ullman（神林訳）; “プログラミング言語ML”, アスキー出版局（1996）

[11] http://www.smlnj.org/

[12] 大野邦夫; “OMGのコンパウンド・ドキュメント標準”, Object World Expo, Tokyo, C14 講演資料（1995）

[13] 大野邦夫;“複合ドキュメント技術への一考察”,情報処理学会研究報告, DD51−7（2005.7）

[14] 大野邦夫, 井上和哉; “xfyによるXBRLの実装”, 画像電子学会第17回VMA研究会資料（2006.7）

[15] 大野邦夫;“文書を構成する型に関する一考察”,情報処理学会研究報告, DD22−1（2000.3）

[16] Cisco; “Application Centric Infrastructure Overview: Implement a Robust Transport Network for Dynamic Workloads”, http://www.cisco.com/c/en/us/products/collateral/cloud.systems.management/aci.fabric.controller/white.paper.c11.729587.html

[17] Cisco; “アプリケーション セントリック インフラストラクチャ（ACI）の原理”, http://www.cisco.com/web/JP/solution/datacenter/literature/white−paper−c11−729906.html

[18] 大野邦夫, 須藤僚, 新麗; “ネットワークコンシェルジュの検討”, 情報処理学会研究報告, DD67−3（2008.7）