デジタル・ドキュメント 22－1
（２０００．３．１７）

文書を構成する型についての一考察

大野邦夫　　吉田正人
ohno@inse.co.jp, yoshidam@inse.co.jp

INSエンジニアリング株式会社
〒141 - 0031 東京都品川区西五反田4 - 31 - 18

最近のXMLの広範囲な分野への適用には目をみはるものがあるが、その考え方やアプローチなどについては問題を感じるものも少なくない。その理由の少なからざるものがDTDに関する理解の困難さや誤解に起因するように思われる。DTDは文字通り文書の型の定義（Document Type Definition）であるが、適用技術や活用手法は極めて不十分である。ここでは、ラッセルのパラドックスに端を発する型の概念と文書の電子化との関係を様々な視点から考察し、現状の到達点と今後の可能性について論じる。

A Study on The Types Which Organize Documents

Kunio Ohno, Masato Yoshida

INS Engineering Corporation
4 - 31 - 18, Nishigotanda, Shinagawa - ku, Tokyo 141 - 0031, Japan

Though the surprising improvement of recent XML technologies, there seem to exist a few problems in their applications. Many of them would be caused by DTDs the specifications of which are rather difficult and easily misunderstood by many people. Although DTD means just document type definition, the methodology to apply or to utilize DTDs is not so clear as another document technologies. This paper surveys and analyses the types which are initiated by the famous Bertrand Russell's Paradox, and describes their current status and future possibilities.

1. まえがき

インターネットによるウェブブラウザの出現以来、電子化文書の世界は革命的とも言えるほどの進展を遂げつつある。その動向は、あらゆる産業分野に浸透し生産活動、経済活動に重大な変化をもたらしつつある。さらに産業分野に留まらず、行政、医療、教育、家庭を含む社会全体に影響を及ぼしつつある。このような広範な社会変化を支える技術革新は、かつて「読み書きソロバン」と言った人間が社会に適応するための基本的な能力の電子化対応を要求するものである。コンピュータの発明はソロバンを駆逐した。電子化文書とそれへのインタラクションは「読み書き」能力を駆逐することは無いが鋭い変更を迫るものであろう。

そのように激しい変化の状況下にあって、この分野の研究者、技術者が思慮すべき観点を押さえておかねばならないであろう。変化が激しい時こそ普遍的な視点を求める必要があると思われるからである。それは何かということに対し独断と偏見を恐れずにあえて申し上げると、それは「型」の概念ではないかと思われる。型という言葉が抽象的であればデータ型の考えを文書や知識の領域にまで拡張するものと考えればよいであろう。

最近のXMLの広範囲な分野への適用には目をみはるものがあるが、その考え方やアプローチなどについては問題を感じるものも少なくない。その理由の少なからざるものがDTDに関する理解の困難さや誤解に起因するように思われる。DTDは文字通り文書の型の定義（Document Type Definition）であるが、適用技術や活用手法は極めて不十分である。ここで論じたいのはまさにこの領域の問題である。

2.用語の定義

2.1 文書とは何か

我々は「文書」という言葉と「ドキュメント」という言葉をほとんど同義で使っているが、それらの定義は必ずしも明確ではない。岩波の「広辞苑（第4版）」[1]によると、「文書」は「文字で人の思想をあらわしたもの。かきもの。ふみ。もんじょ。」とのことであり、「ドキュメント」は「記録。文献。」とのことである。別の視点から、和英辞典（三省堂　『GLOBAL&CENTURY英和・和英辞典』）[2]の和英の部で「文書」を調べてみたところ、「【書類】papers;（資料・証拠としての）a document（→書類）;【通信文】a letter, correspondence;【記録】a record」とのことで、英語の概念の多様さが推察される。次に英和の部で「document」を引いてみると、「文書[証拠書類, 公的記録, 証書など]」ということであった。このことから、「文書」に関しては、日本語では書類や記録といった程度の概念であり、それ以上の掘り下げは難しいようであるが、英語の世界は書類、証拠、通信文、記録といったサブカテゴリで代表される豊富な概念を持っている。さらに英語の世界の「document」の概念を把握するために、2種類の類義語辞書を調べてみた。

J. I. Rodale; The Synonym Finder, Warner Books, (1986)によると、「official paper, certificate, charter, record, chronicle 」ということである。Roget's Thesaurus, St. Martin's (1962)では「evidence 466n, corroborate 466vb, demonstrate 478vb, record 548n」ということで、名詞としての意味で2種類（evidence, record）、動詞としての意味で2種類（corroborate, demonstrate）が提示されている。なお個々の意味記述に後続する数字は、意味カテゴリの系列番号で、その後のnは名詞、vbは動詞を意味している。

英語の「ドキュメント」は、日本語の「文書」に比べるとより具体的かつ厳密な概念であり、「証拠を提示するもの」という意味と、「過去の事実の正式な記録」という意味で用いられていると言える。この2種類の意味は、情報メディアが保有する情報の伝達と記録という二重の性質を代表している。

ところで、以上の議論の背景は、文書が「書類」の同義語に近い紙の文書の場合である。電子化された文書の場合は、表示または印刷された文書の背後に電子化されたデータによる構造が存在する。その構造は文字コード、図形データ、画像データ、数値データなど様々なデータが関与している。これらをどのように関連付け、有効に使えるようにするかが、電子化文書の進歩を促すアプローチであろう。それでは、そのような電子化文書はどのように構築すれば良いか。

回答は存在する。巨大な構築物と言えどもそれらは部品から構成される。電子化文書の場合も信頼性の高い標準的な部品を用い、標準的な工法を用いて構築すれば良いのである。そのような考え方は、コンピュータシステムの構成やプログラム言語の世界では検討されており、「ソフトウエア工学」という分野が確立されている。システムのモジュール化、再利用などと言うキーワードはこの世界から生まれてきた。

モジュール化、再利用を標榜する分野としてオブジェクト指向が取り上げられ、オブジェクト技術の標準化を指向するコンソーシアムとしてOMG（Object Management Group）が設立されたのもこの流れに沿うものである。この考え方を適用し、文書も部品から構築されるシステムであるという考え方を徹底させることが基本であろう。ところで、文書を部品から構成されるシステムとして捉えてゆくアプローチにも2種類の考え方がある。一つは、最も基本的な部品として、現状のプログラミング言語が用いている型を流用するものである。もう一つは、「文字で人の思想をあらわしたもの」という広辞苑の記述に相当する「意味レベルの部品」を検討するアプローチである。後者は、機械翻訳などの自然言語処理分野の領域であるが、分析哲学や論理学からのアプローチも考えられて然るべきであろう。ナレッジマネジメントなどへの適用も今後は期待される。

部品を型として定義してゆきたいというのが、本資料の基本的な立場であるが、その理由は他の構築物の場合と同様である。型により実体（インスタンス）を量産し、再利用可能とすることである。問題は型をどのようにして決めるかである。結論的には、明確にそれ自体を定義するか帰納的なプロセスを通じて分類してゆくかの何れかであろう。要するに集合を、内包的に定義するか外延的に定義するかのアプローチである。

2.2 型とは何か、

それでは型とは何か。型は英語のtypeを意味する。三省堂 『GLOBAL&CENTURY英和・和英辞典』で引いてみると、「型, タイプ, 類型, 種類,（顕著な共通点によって客観的に区別される型, 種類）」と記述されてい

る。「顕著な共通点によって客観的に区別される」という概念を持つことから、集合の概念に関係することが推察できる。

2.2.1 ラッセルのパラドックス

集合論と関係する型の考え方は、バートランドラッセルの数理哲学に端を発している。彼の型の理論は、「ラッセルのパラドックス」として有名な自己参照を含む集合の定義に潜む本質的な矛盾を回避するための一つの考え方で、これについては、"The Principles of Mathematics - Appendix B: The Doctrine of Types,"[3]に記されているものが原典である。しかし、これは難解で、むしろ、彼の晩年の著作である"My Philosophical Development,"[4]の記述が分かりやすいので紹介する。それによると「We shall have to distinguish between propositions that refer to some totality of propositions and propositions that do not. Those that refer to some totality of propositions can never be members of that totality.[4]」ということで、「我々は、命題の全体を指示する命題とそういうものを指示しない命題とを区別せねばならないであろう。そして命題の何らかの全体を指示するところの命題は、決してその全体の要素ではありえないのである。[5]」次に、「We may define first - order propositions as those referring to no totality of propositions; second - order propositions, as those referring to totalities of first - order propositions; and so on, ad infinitum.[4]」ということで、「我々は、命題の全体を指示することのない命題を第一階の命題と呼び、さらに第一階の命題の全体の指示する命題を第二階の命題と呼び、そのようにして無限に進むことができる。[5]」という手法により、矛盾を回避する。ラッセルの定義では、型とは上記に基づく命題の集合である。

2.2.2 データ型

計算機プログラムに関係する型は、プログラム言語の発展と共に進展した。すなわち、整数、浮動小数、文字、ブーリアンなど、プログラム言語のデータ型である。C.J.Cleaveland著、"An Introduction to Data TYPES" [6]（日本語訳、小林光夫訳、"データ型序説"共立出版[7]）という本にデータ型を中心にした型について解説されている。「型とは値の集合である」というのがプログラム言語でのデータ型原始的な定義であったようであるが、JimMorrisという人が「型は集合ではない」という論文を書いて以来、種々の議論が起こったと言う（データ型論争）。1970年代に入って抽象データ型の概念が普及してからは、「型とは、値の集合およびその上の演算の集合である」というのが定説になっている。なおラッセルの定義において、「命題」を「演算」と読み換えれば、データ型の定義に近いものになる。

2.2.3 部分型と継承

型は集合と同様にして、包含関係が成り立つ場合がある。ある型としての条件を満たしながら、さらに別の条件を満たす場合、その型はもとの型の部分型（subtype）である。部分型はもとの型を「継承する」という言い方も可能である[8]。

3. 型概念の拡張

3.1 オブジェクト技術

型概念の拡張という意味で、最も自然なものは、多くのオブジェクト指向プログラミング言語において定義されるクラスである。一般のクラス定義においては、状態（インスタンス変数）と振る舞い（メソッド）が定義される。状態は静的なデータを表し、振る舞いは動的な挙動を表す。ある種のクラス定義では、クラス全体の属性（クラス変数）が定義されることもある。クラスも型と同様に継承することが可能で、継承されたクラスをサブクラスと呼ぶ。

3.1.1 Smalltalk

オブジェクト指向言語のパイオニアは、1970年代にXerox PARC（ゼロックス、パロアルト研究センタ）で開発されたSmalltalkである[9]。Smalltalkは、全てがオブジェクトで構成されるため型を持たない。その代り多様なクラスを持つ。数字や文字は当然、図形、画像、音声といったクラスを持ち、そのインスタンスの生成、編集、実行といったメソッドを持つ。さらにMVC（Model View Control）メタファ[10]というその管理の枠組みも提供していた。この概念は、当初は、オブジェクトへの入出力のためのプログラミング手法として理解されていたが、その後、オブジェクト分析設計、ヒューマンインタフェース、アクティブドキュメントなどの幅広い分野に関係する普遍性を持つことが分かってきた。

3.1.2 汎用オブジェクトモデル

型の定義や継承（クラス）の概念がプログラム言語の中だけでしか使えないのは不便である。プログラム言語独立なニュートラルなオブジェクトや型を定義してもよいのではないか。そのような観点を含めた汎用オブジェクトモデルという概念が生まれた[8]。それによると、オブジェクトへのメッセージ構文は、入力パラメタを出力パラメタに対応付ける代数学的なオペレーションとして位置づけられる。

$$\Omega : (x_1: \sigma_1, x_2: \sigma_2, \ldots\ldots x_n: \sigma_n) \rightarrow (y_1: \rho_1, y_2: \rho_2, \ldots\ldots y_m: \rho_m)$$

ここで、Ω：オペレーション名、x_i：型σ_iのパラメタ、y_j：型ρ_jの結果である。

最近、異機種分散環境の構築ツールとして普及しつつあるOMG（Object Management Group）のCORBA（Common Object Request Broker Architecture）のコア技術であるIDL（Interface Definition Language）の構文と型は、このモデルに基づいている。この詳細な位置づけは、http://www.omg.orgにおけるOMG Document 90 - 9 - 1 "Object Management Architecture Gude"に記されている[11]。

3.2 データベース

オブジェクトが拡張された型であるなら、データベースは、型を厳格に管理する容器である。従って、文書における情報の「記録」という性質を代表するものとして対応付けられるであろう。データベース自体を型またはクラスと考えることも検討されて然るべきであろう。

3.2.1 RDB

RDBは関係（リレーショナル）モデルに準拠するデータベースである。関係モデルはレコード（属性と値のペア）の集合としてのテーブル（表）で定義され、モデルの操作は、以下に5つの関係代数演算で定義される。

- 選択（selection）
- 射影（projection）
- 直積（product）
- 集合和（union）
- 集合差（set - difference）

結合（join）は、基本演算ではないが、複数の表を統合するデータ操作として用いられる。SQLは、関係代数演算を実現する標準の照会言語（Standard Query Language）で、select, from, whereという3つの節で構成される。レコードにおけるフィールドのデータは、関係演算が扱える値のデータ型に限定される。RDB自体を一つの型として扱う考え方もあり得る。その場合の値は個々のレコードであり振る舞いは、SQLを構成する関係代数演算となるであろう。

3.2.2 OODB

OODBは、当初オブジェクト指向プログラミングにおけるインスタンス永続化のツールとして出発した。そのためSQLのような言語独立なインタフェースでなく、OO言語（殆どがC++）に準拠したインタフェースであった。やがて個々のOODBはOQL（Object Query Language）という照会言語を持つようになったが、DB間の互換性は無かった。その標準化を目指してODMG[12]というコンソーシアムが設立され、ODMG93という規格が取りまとめられたが、（このコア規格はODLと呼ばれるインタフェース定義言語で、CORBAのIDLのスーパーセットのようなものであった）急速に進展したRDBに比べ、OODB自体の市場ニーズが顕在化しなかった。但し、OODBは、データ型を拡張し、マルチメディアなどが扱える点に大きな特長があった。OODB自体を一つの型として扱う考え方もあり得る。その場合の枠組みは、通常OODBを扱うOOプログラミング環境をそのままサポートするものとなる。すなわちOOプログラミングで定義された型とそのインスタンスを永続的に蓄積することが、OODBの基本的な用途だからである。OODB独自の照会言語（OQL）を含む型として扱う方法も考えられるであろう。

3.2.3 ORDB

OODBに並行してORDB（Object Relational DB）が開発された。ORDBは、SQLを用いながら扱うデータ型を拡張可能とした点に大きな特徴がある。ORDB自体を型として扱う考え方もあり得るであろう。その枠組みは、RDBを型として扱う場合の拡張となる。その値は、個々のレコード（拡張された型の値を含む）であり、振る舞いはOO拡張されたSQLであるSQL3を構成する関係代数演算である。

4. 文書概念の拡張

4.1 文書ライフサイクル

文書が「情報の伝達と記録」という二重の性質を持つメディアであることは、すでに述べたとおりであるが、それに関連して「文書のライフサイクル」も重要な概念である。一般的に公的な文書は、作成、修正、承認、配布、保管、保存、廃棄といったライフサイクルを辿る。文書における情報の伝達という役割は、上記の配布フェーズにフォーカスを当てた場合の役割である。情報の記録というという役割は、上記の保管、保存フェーズにフォーカスを当てた場合の役割である。ライフサイクルの概念は、公的な組織としての文書の作成、配布、管理に関係する。従って文書が組織としての意思決定プロセスや責任者を反映したものとなるが、これはまえがきで欧米のドキュメントが「証拠を提示するもの」という意味と、「過去の事実の正式な記録」という意味で用いられることに照応する。

文書の電子化は、ライフサイクル的に見ると、作成・修正プロセスから開始された。それは、プログラム作成用のエディタとして以前から計算機システムに組み込まれていたツールから容易に転用できたからである。従って、作成フェーズだけが電子化され、印刷された紙による承認、配布、保管といったライフサイクルでの運用が行われた。電子化文書の配布、保管、保存は、ネットワーク技術やCDROMのような媒体に依存すると同時に適格な参照ツールが必要となる。アドビのアクロバット・リーダがDTPシステムで開発されたドキュメントのビュワーとして開発されたが、開発当初は必ずしも普及しなかった。配布、参照フェーズが普及したのは、インターネットによるメールの配信やウェブホームページへの参照が可能になってからである。

保管、保存フェーズの電子化は、文書データベースの構築に関するテーマであるが、今後の検討を要する課題である。電子化文書の承認フェーズの問題は、ワークフロー管理の自動化を含むテーマであるがやはり今後の課題である。

4.2 作成編集文書モデルの型

4,2,1 テキストエディタ

作成・編集文書型の基本となるモデルはテキストエディタであろう。コンピュータの発明後、プログラム言語の誕生とほとんど同時に文書の作成、編集プロセスは電子化された。当初はTSSシステムにおけるテキストエディタとして主にプログラムの作成、編集に用いられた。テキストエディタも行エディタから画面エディタへと進化し、Emacsやviのように、現在でも使われているものもある。

4.2.2 ワードプロセッサ

その後、マイクロプロセッサの普及と高性能化に伴い、スタンドアローンのワードプロセッサとして、またはマイクロコンピュータのワープロアプリケーションとして、文書作成・編集の電子化が実現された。電子化された文書は、紙の文書に比べ優れた特徴を持っている。例えば文字列一致による検索機能やグローバル・リプレース機能である。これらは紙の世界では到底実現不可能なものであった。ワードプロセッサは、テキストエディタに比べ印刷機能を保有している。そのために、用紙サイズ、ページあたり行数、行あたり文字数、段組、マージンなどの設定のような簡易なページレイアウト機能を保有している。これらはテキストエディタのオプ

ショナルな機能というよりは、印刷文書の属性の設定という観点で捉えるべきものであろう。

4.2.3 Xerox PARCの成果

Xerox PARCは、1970年代に現在のパーソナルコンピュータの原点となったALTOマシンを開発した。その狙いは、LRG（Learning Research Group）の責任者であったアラン・ケイの言葉を借りると「ダイナブック」と呼ばれるもので、パーソナルコンピュータを「動的な本」というメタファで捉え、関連する技術を実現しようとするものであった[13]。技術戦略的には、センターの大型計算機に機能を集中させるTSSの限界を突破することにあった。すなわち、大型計算機の性能・機能を個々のパーソナルコンピュータやサーバマシン上に分散させることにより、使いやすいコンピュータ環境を実現することであった。コンピュータの操作には、GUI（Graphical User Interface）が用いられた。そのため、マウス、ビットマップ・ディスプレイ、マルチウインドウ、ローカルネットワークと言った革新的な技術が採用された。その当時の文書作成編集環境のターゲットは、WYSIWYG（What You See Is What You Get）であった。WYSIWYGとは、画面で見たとおりに印刷されるという意味である。そのために、マルチフォントをサポートした、文字、図形、画像を含む文書処理、レイアウト処理を実現した高機能なエディタが実現された。ALTOマシンの後継機種として、ドラド、ドルフィンといった専用リスプマシンが開発された。ドルフィンと同一ハードを用いた文書処理専用のStarワークステーションも誕生した。Starワークステーションの文書環境は、WYSIWYGを実現した最初の商品であった。

4.2.4 DTP

パロアルト研究センタの技術はアップルコンピュータのマッキントッシュによりコンシューマ向けの商品として普及した。時を同じくして、アドビのPostScript、キャノンの卓上レーザプリンタが商品化され、アルダスのPageMakerを代表とするデスクトップパブリッシング（DTP）の全盛時代となった。DTPの波はUNIXワークステーションにも波及した。Interleaf、Framemakerといった高性能、高機能なDTPが開発され、オフィスはもとよりエンジニアリング分野でも使われはじめた。

テキストエディタとDTPの外見的な相違を列挙してみる。

(1) テキストエディタが、データ型として文字と文字列しか扱わないのに対し、DTPは、文字、文字列以外にも、図形や画像、さらにはそれらを記述するための枠（フレーム）といった情報を扱う。

(2) テキストエディタでは、データのアクセス単位は、行数と文字数といった単純なものであるが、DTPは、ページやコラムといったレイアウト情報がさらに付随する。

(3) テキストエディタの文字、文字列は、属性を持たないが、DTPの場合はフォント、サイズといった属性を持つ。

要するに、テキストエディタが扱う文書データは、単一の単純な文字の列（テキストファイル）であるのに対し、DTPは、多数の要素から構成される構築物（システム）となっている。この構築物に対して、種々の捉えかたが可能であり、それが、今日のデジタルドキュメント技術を構成している。

4.3 マークアップ言語

4.3.1 nroffとtroff

マークアップ言語の発端は、単純なテキストエディタを用いて、専用のワードプロセッサが作成するような体裁の良い文書を作るためのツールであった。UNIX上のnroffはそのためのポピュラーなツールであった。日本語用に機能拡張されたjroffというツールも開発され、1970年代から80年代のはじめにUNIXユーザにより用いられた。troffは、写植印刷向けのマークアップ言語である。これらの言語は、基本的なコマンドを組み合わせた複合的な機能をマクロとして定義することも可能であり、そのようなライブラリを用いて統一された体裁の文書を作成することが可能である。

4.3.2 TeX

TeXは、ドナルド・クヌース教授が、数式を含む数学分野の論文作成支援用に作られたマークアップ言語である。TeXは、文字フォントを定義するツールやベクター図形を記述するツールを保有しているため、アカデミックな世界のみならず、印刷出版分野で幅広く用いられた。TeXもマクロを定義することが可能で、種々のマクロライブラリが存在する。LaTeXもその一つである。

4.3.3 SGML

以上、nroff、troff、TeXといった種々のマークアップ言語が登場し、各々のツールが勝手な拡張を遂げる中で、拡張メカニズムを保有する汎用マークアップ言語（GML）を作るアイデアが生まれ、さらにこれがISOで標準化されSGMLとなった。先に述べたオブジェクト指向言語における汎用オブジェクトモデルのようなものである。オブジェクトの世界で用いられたクラスの概念に相当するものとして、DTD（Document Type Definition）という概念を導入し、個々の文書をそのインスタンスとして定義付けるた。

DTDは文書を要素の集合とみなす。要素は基本的にはテキストで定義され、その属性を定義することも可能である。さらに実体参照のためのパラメタを定義し、繰り返し参照される情報や外部ファイルなどを参照する。

一方、SGMLのDTDに関しては、分かりにくいという批判がこの10年にわたり継続的に存在してきた。この批判は2種類のものが混在しているように思われる。一つは、メタという概念そのものへの批判である。この批判の主の多くは、個別マークアップ言語の世界にいた人たちである。「TeXならDTDなんて定義する必要もないのに、なぜ必要なのか？」と言ったものが代表である。「個別マークアップと汎用マークアップの区別も知らないのか！」というのが多くの場合のSGML側からの不親切な回答であったが、そのような不遜な態度をとり続けた結果、自らの個別DTDによる個別マークアップ言語であるHTMLの爆発的な発展によりしっぺ返しを受けたように思える。DTDに対してはさらにもう一種類の批判勢力が存在する。それは、DTDの構文と表記法に関するものである。XMLスキーマを期待している人々がその代表である。DTDとオブジェクト指向技術のクラス定義とはメタ

な枠組みを定義するという意味では同じであるが、定義の仕方は雲泥の相違である。継承を陽に使えるようにすることなどが先ず第一に要請される。DTDをクラス的な考え方で定義してもらえないかという要求こそXMLスキーマに他ならないが、この要望は以前から存在した。その実現は決して難しい話ではない。

例えばプログラム言語の枠組みの中で、DTDで定義された文書をそのDTDに対応した文書クラスのインスタンスとして定義することは可能である。現にハイエンドのDTPシステムであったInterleaf5では、編集対象の文書はカスタマイズ用言語のInterleaf Lispの文書クラスのインスタンスとして定義された。一般のオブジェクトのインスタンスとの相違は、インスタンスが拡張可能なツリー構造となるため、エレメントツリーを移動するためのget - first - child, get - last - child, get - next, get - previous, get - parentといったナビゲーション・メソッドが提供されていたことであろう[14]。対象とする個々のエレメントに辿り着けば、そのエレメントの値などへの操作は、一般のオブジェクトのインスタンス変数の操作の場合と何ら変わらない。

今後XMLスキーマが順調に勧告化されるかどうかは不明であるが、その期待は大きいものがある。ただ現在のXMLスキーマ仕様はあまりに膨大で、たとえ勧告化されても使いこなせるものかどうかが若干危ぶまれる[15]。

4.4 デジタルドキュメント

4.4.1 マルチメディア

かつて、パソコンの表示出力技術として映像を中心としたマルチメディアがブームとなったが、現在はやや鎮静化している。マルチメディアの特徴は人間の視聴覚を活用する優れた認知性にある。文字や静止画に比べると、動きを伴った映像や音声は事実を知らせる手段としては説得力がある。しかし高品質な映像の制作コストは高価であり、営業のためのプレゼンテーション、または教育、娯楽などのような配布先が多く単位コストが低い分野に限られる。また、抽象的な内容を議論・伝達するような場合には、文字というメディアは有力であり必須であろう[16]。したがって広辞苑における文書の説明である「文字で人の思想をあらわしたもの。」という観点からすると、マルチメディアは非文書的メディアである。

4.4.2 アクティブドキュメント

先にDTPの項で述べた、図形や画像を含む文書において、動的な図形（アニメーション）や画像（映像）を扱い、そのスナップショットをフレームに固定することが可能な文書はアクティブドキュメントと呼ばれる。この概念は、事実に対する説得力が強いマルチメディアと抽象的な論理記述に優れる文字メディアの文書を融合したもので、見方を変えるとPARCのダイナブックのコンセプトを実現したものと言えるであろう。

1990年に登場したインターリーフ社のDTPソフト、Interleaf5は、その商品名としてアクティブドキュメントという言葉を用いた[17]。Interleaf5の場合は、SGMLの項で述べたInterleaf Lispというカスタマイズ用の強力なLisp処理系がDTPの背後で走行しており、ユーザやネットワークからのイベントを常に監視している。従って、ユーザの操作により、CAD、CASEツール、スプレッドシート、アニメーション、映像など、他のアプリケーションを起動し、その結果をフレームに貼り付けたり、常時走行するアプリケーションのプロセスが、Interleaf5を起動し、それの特定のドキュメントを開き、さらに、先に説明したエレメントツリーのナビゲーションを通じてその特定のエレメントにアクセスしデータを書き込み、その文書をセーブするといった処理が可能であった[14]。

4.4.3 複合ドキュメント

その後、類似のメカニズムがパソコン上の「複合ドキュメント」として提供されるようになった。WindowsのOLE（Object Linking & Enbedding）とアップル、IBMなどのOpenDOCである[18]。OLEは、Windows環境において、「ワード」のような文書に、Windows上の他のアプリケーション結果やスナップショットを貼り付ける機能を提供する。その場合の他のアプリケーションは、Windows環境におけるオブジェクト（COM）として位置付けられる。OpenDOCの場合は、さらに意欲的な枠組みを定義した。文書というものを文書要素を格納するコンテナとして定義し、各文書要素はパーツとして位置付けられる。さらにパーツは、その要素のコンテンツを作成編集可能なエディタまたはその要素コンテンツの参照機能を提供するビュワーとして定義される。パーツは、再帰的にパーツを要素とすることが可能になっている。

OMGはOpenDOCの枠組みをIDL化し、CORBAファシリティの要である「複合ドキュメント標準」とした。しかしながらこの規格は普及せずに終わった。その理由は、パソコン上の文書作成管理ソフトとして、MSのワードが事実上の標準となり複合ドキュメントとしてはOLEに軍配が上がったためである。さらに次項で述べるHTMLによるインターネットのウェブが普及したことにより複合ドキュメントに代わる簡易ネットワーク・ドキュメントとでも呼ぶべき新たなインフラが展開し始めたことにもよる[19]。

4.4.4 HTML

インターネットという脆弱だが基本的にフリーなインフラ上で、容易に配布・参照可能な文書形式としてHTMLが考案された。これは表示に特化されたSGMLアプリケーションである。HTMLにとって幸いだったことに、参照ツール（ビュワー・ブラウザ）が無料で提供された。その結果、インターネットの普及とともに、利用者は爆発的に増大した。HTMLが従来の電子化文書と異なる特色は、配布・公開するメカニズムを極めて安価なコストで実現したことにある。これは、HTMLの技術的な特徴というよりは、一挙に普及拡大したことによる、事実上の標準化（デファクト・スタンダード）を達成したことにある。

4.4.5 XML

XMLについては、もはや解説の必要はないと思われるが、型との関係については後で述べる。

4.5 配布・保管文書モデルの型

4.5.1 Interleaf RDM

文書のライフサイクル管理を取り上げたシステムとしては、Documentum、PCDocs、Livelinkと言った製品が出ているが、これらに先行してInterleaf社のRDM（Relational Document Manager）が製品化されていた[20]。RDMは、その言葉から推察できるとおり、RDBを用いて文書とワークフローを管理検索するシステムであった。

RDMの文書は、InterleafのDTPシステムで作成され、それは図書館の図書カードのような管理票でもって厳格に管理するメカニズムを提供していた。たとえば、作成者、作成日時、レビュー担当者、その受け付け日時、完了日時、承認者、承認日時などが、個々の文書についてRDBの個別のフィールドで管理されていた。我々は、RDMを用いて、ISO9000シリーズの文書管理を行うモデルシステムを構築したことがある[21]。この枠組みや機能は優れたものであったが、業界標準にはならなかった。その理由は、SGMLに準拠していなかったためと言われ、その後ベンダーもSGML化に向けて努力を続けた。

Interleaf社はその後、Interleaf5用のSGMLツールキットを開発・提供し、一部のユーザはそれを利用し始めたのであるが、Interleaf Lispの性能やカスタマイズ機能の不十分さに起因し、日本では実用的に使えるものにはならなかった。

4.5.2 SGMLデータブレード

前項で述べたとおり、1995年ころまでは、ワークフロー管理を含む電子化文書の枠組みをSGMLをベースに検討していたのであるが、時代の要請もあってマルチメディアを包含するSGML文書管理を検討しはじめた。このコンセプトは、最近「クロスメディア」と呼ばれているコンテンツ管理思想に近いものである

SGMLは、基本的な型としてはテキストデータしか扱わないので、図形、画像、マルチメディアといった情報は外部エンティティとして扱われる。しかしこれらのデータを外部ファイルとしてでなく、データベースで管理されるコンテンツとして扱い、厳格な履歴管理を行うというのが開発当初のねらいであった。ORDBのILLUSTRAがその用途としては適格であると思われたので、SGMLデータ管理と図形、画像、映像、音声などの管理をILLUSTRAで行うことを検討した。なお、マルチメディア関連の管理ツールは、ILLUSTRAがデータブレードとして保有していた。従ってこれらに関しては、そのデータブレードをそのまま活用する方式を採用した。一方、SGML管理に関しては、自己開発した。詳細は別の文献[22]に記されているので参照して頂きたい。

SGMLデータブレードのコンセプトは、文書の要素というものを、ORDBとそのミドルウエアであるデータブレードで実装した点にある[23]。文書自体はクライアントアプリケーションで、エレメントツリーをナビゲートし、個々のエレメントにアクセスする。外部エンティティとして定義される画像、図形、映像、音声などは、それらを外部エンティティとするエレメントから専用のデータブレードのAPI経由で起動され、サポートされているビュワーで参照する。

このシステムは、アーカイブ化されたマルチメディア・ドキュメントの検索・参照システムということが可能である。最近XMLによる商品カタログ管理などで話題になっているコンテンツ管理システムのSGML/ORDB版と位置付ける事が可能である。

4.5.3 SGML/XMLデータカートリッジ

ORDB版に続いて、標準のRDBであるOracle版の開発を行った[24]。マルチメディアなどに関しては、DB管理の対象にならないので、外部エンティティのファイルで扱うことにした。SGMLデータと関連ファイルの整合を完璧にするために、文書インスタンスもサーバ側で管理することになった。技術的には退歩であるが、実用価値は向上し、かなりの数のシステムが出荷され現在も運用されている[25][26]。

本システムの開発時にXMLの規格が議論されていたこともあり、SGMLのサブセットであるXMLへの対応（終了タグを省略せずに挿入する等）を考慮したが、具体的なニーズは、すでに決められたDTDによるユーザからのものがほとんどである。

4.5.4 SGMLエレメント

SGML/XMLデータカートリッジは、幅広い分野の技術ドキュメントや社内規定文書の検索・参照用に用いられている。参照用のビュワーとしては、SGMLブラウザが標準であるが、図面や画像の表示用にAcrobatリーダやJPEG,GIF（ウエブブラウザによる）などが用いられている。インターネット経由で参照される場合は、実時間でHTML変換されウエブブラウザに送られる。

SGMLデータブレード、SGML/XMLデータカートリッジで追求したコンセプトは、3層クライアント・サーバ・システムで実現する文書モデルであった。これを、MVCモデル的に考察すると、基本モデルをSGMLのエレメントツリーで定義し、そのデータ管理にORDBまたはRDBを用いた。参照用のビューとして、SGMLブラウザまたはHTMLブラウザを用い、コントロールとしては、ミドルウエアのAPIによるSGMLツリーの操作を対応つけることができる。

以上のようにコア的なモデルのMVCは出来上がっているので、次の段階としては、応用レベルのモデルの型であるDTDの問題である。データカートリッジで用いられているSGML DTDは、個別の専用的なものであるが、ライフサイクル管理用のタグも定義されている（作成者、承認者、承認日時、改定履歴など）。本来的には、このようなエレメントの共有化、標準化の進展を期待したいところであった。

4.5.5 XMLエレメント

以上のような経緯で分かるとおり、SGMLのDTDにおける共通エレメントの標準化はなかなか進まないのであるが、XMLになって若干動きが出てきた。前項のわれわれのSGMLモデルに対応する概念と等価なものとして、DOM（Document Object Model）のツリーが定義された。ビューは、XMLブラウザである。コントロールはDOMのAPIである。DOMのAPIは、OMGのIDLで定義されたため、プログラム言語から独立したAPI仕様となっている。このようにして、XMLのコア的なモデルは確立した。次のレベルとして、DTDやXMLスキーマで定義されるべき型が出現すべきであろう。このレベルは、業界横断的（Horizontal）なものと業界依存的

（Vertical）なものとに区分されるべきであろう。前者の例としては、例えば、電子ビジネスカードの規格であるvCard[27]や、スケジュール管理規格のiCalendar[28]などが挙げられる。これらの規格は、IETFにおいて、オブジェクトモデルとして定義され、さらにXMLのDTDとしても定義されつつある[29][30]。これらの規格が注目に値するのは、MSのOutlookやロータスのOrganizerが、これらの規格に準拠したデータでシステムを構築していることが挙げられる。

地図情報に関するオブジェクトモデルとDTDも提案されつつある。データベース振興センタが取り組んでいるG-XMLである。現在はα版であるが、種々のアプリケーションとの相互運用に成功すれば将来は幅広い分野で使われるようになるであろう。

5. あとがき

以上、文書を構成する型について、日ごろ考えていたことを書いたのであるが、XMLの技術的な進展のスピードに比べ、文書系の規格化の遅さは気になるところである。その要因として、型の問題が潜んでいると思われる。ラッセルが型についての最初の考察を行って一世紀が過ぎているが、SGML、XML、OMG IDLなどの標準化動向を見る限りにおいて体系立った進歩はしていないと見ることもできる。しかし、コンピュータが誕生し、コンピュータ上で型を定義し、操作できるようになっただけでも大きな進歩と言えるのかもしれない。

Rubyで構築したKMサーバ[31]の開発において、レガシーシステムのデータをXMLのタグ情報で対応付けたのであったが、もしも、これらの情報が標準化されていれば、対応付けは極めて容易になる。対応付ける情報は、人、日時、所番地、企業名、電話番号など、様々であるが、これらにvCard、iCalendar、G-XMLなどのエレメントでレガシデータが構築されていれば、相互運用は極めて容易になるはずである。

ただし、このような活動は、技術だけではなく業界政治学的なアプローチが必要となる。そのためには、幅広い技術分野とグローバルな市場ニーズ、国際動向、社会動向を知りつつ、標準化へのリーダシップを取って行ける人材が必要となるであろう。併せて文書を構成する型について、本格的に取り組む研究を期待したい。

文献

[1] CD-ROM版広辞苑, 岩波書店, (1998)

[2] CD-ROM版GLOBAL&CENTURY英和・和英辞典, 三省堂（1999）

[3] B. Russell; "The Principles of Mathematics", Routledge, London, (1903)

[4] B. Russell; "My Philosophical Development", G. Allen & Unwin, (1959)

[5] B.ラッセル, （野田訳）; "私の哲学の発展",バートランドラッセル著作集（別巻）, みすず書房, (1962)

[6] C.J.Cleaveland; "An Introduction to Data TYPES", Addison-Wesley, (1986)

[7] C.J.Cleaveland, (小林光夫訳); "データ型序説", 共立出版, (1990)

[8] P. Wegner; "The Object-Oriented Classification Paradigm", Research Directions in Object-Oriented Programming edited by Bruce Shriver & Peter Wegner, MIT Press, (1987)

[9] A. Goldberg, D. Robson; "Smalltalk-80", Addison-Wesley, (1983)

[10] 竹内, 梅村 ; "Smalltalk入門", サイエンス社, pp.93-95, (1986)

[11] Object Management Architecture Guide, Rev.2.0, OMG TC Document 92.11.1 (1992)

[12] M.E.S.ルーミス,(野口訳); "オブジェクトデータベースのエッセンス", アジソン・ウェスレイ・トッパン, pp.93-119, (1996)

[13] 大野 ; "情報の入出力方法と操作性", 電子通信学会誌, Vol.87, No.4, pp362-367, (1984)

[14] 大野 ; "Interleaf5におけるオブジェクト指向(2)", オブジェクトレポート, Vol.1, No.2, 創研プランニング, pp.15-17, (1992)

[15] http://www.w3.org/TR/2000/WD-xmlschema-2-20000225/

[16] 大野 ; "テクニカルコミュニケーションと情報メディア", 情報処理学会第一回テクニカルコミュニケーションシンポジウム講演資料, (1993)

[17] 大石; "オーバービューオブ Interleaf5", Super ASCII, Vol.3, No.10, (1992)

[18] 大野; "OMGのコンパウンドドキュメント標準", Object World Expo / Tokyo シンポジウム講演資料, (1995)

[19] 河込,中村,大野,飯島; "分散オブジェクトコンピューティング", 共立出版, (1999)

[20] 大石; "オーバービューオブ Interleaf5", Super ASCII, Vol.3, No.11, (1992)

[21] 中島, 大野 ; "ISO9000シリーズ用の電子化文書管理システム", 情報処理学会テクニカルコミュニケーション研究グループ研究報告, (1994.9)

[22] 大野, 佐藤 ; "ORDBによるマルチメディア・ドキュメントの管理", 情報処理学会デジタルドキュメント研究会研究報告 DD7-5 (1997.5)

[23] 大野邦夫; "ミドルウエアによるSGML/XML文書管理の枠組みの検討 - ネットワークとドキュメントのオブジェクト技術による融合", 情報処理学会デジタルドキュメントシンポジウム'98論文集, pp.57-66 (1998.1)

[24] K.Ohno, M. Beyer; "Development of SGML/XML Middleware Component", Conference Proc. on SGML/XML Europe'98, pp.373-382

[25] 矢島, 藤津, 大野 ; "SGMLデータカートリッジによる文書管理システムの構築",情報処理学会デジタルドキュメント研究会研究報告, DD14-2 (1998.9)

[26] 加藤,菊地,宮澤 ; "SGMLによる銀行マニュアルの効率的な作成と配布", 情報処理学会デジタルドキュメント研究会研究報告, DD21-2 (1999.11)

[27] F. Dawson & T. Howes; "vCard MIME Directory Profile", IETF vCard仕様 (RFC2426), (1998)

[28] F. Dawson & D. Stenerson; "Internet Calendaring and Scheduling Core Object Specification", IETF iCalendar仕様 (RFC2445), (1998)

[29] Frank Dawson; "The vCard v3.0 XML DTD", Internet Draft draft-dawson-vcard-xml-dtd-03.txt (1998)

[30] F. Dawson, S. Reddy, & D. Royer; "The iCalendar DTD Document", Internet Draft draft-many-ical-xmldoc-01.txt, (1999)

[31] 吉田,大野,藤田,前,廣瀬 ; "オブジェクト指向スクリプト言語RubyによるXML応用システムの検討", 情報処理学会デジタルドキュメント研究会研究報告, DD21-3 (1999.11)