(19)日本国特許庁(JP)　(12)公表特許公報(A)　(11)特許出願公表番号
特表2007-532986
(P2007-532986A)
(43)公表日　平成19年11月15日(2007.11.15)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 17/21	(2006.01)	G06F	17/21	570L	5B009
		G06F	17/21	501T	

審査請求 未請求　予備審査請求 未請求　（全 49 頁）

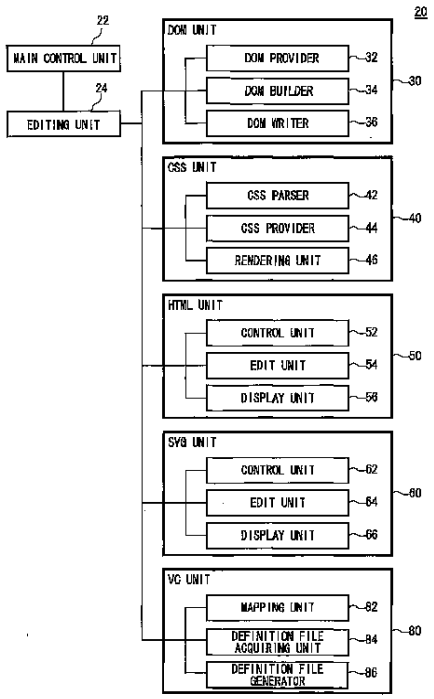
(21)出願番号	特願2006-535891 (P2006-535891)
(86)(22)出願日	平成17年4月8日 (2005.4.8)
(85)翻訳文提出日	平成18年12月6日 (2006.12.6)
(86)国際出願番号	PCT/JP2005/007287
(87)国際公開番号	WO2005/098664
(87)国際公開日	平成17年10月20日 (2005.10.20)
(31)優先権主張番号	60/592,369
(32)優先日	平成16年8月2日 (2004.8.2)
(33)優先権主張国	米国 (US)
(31)優先権主張番号	特願2004-114524 (P2004-114524)
(32)優先日	平成16年4月8日 (2004.4.8)
(33)優先権主張国	日本国 (JP)
(31)優先権主張番号	特願2005-20457 (P2005-20457)
(32)優先日	平成17年1月27日 (2005.1.27)
(33)優先権主張国	日本国 (JP)

(71)出願人　390024350
株式会社ジャストシステム
徳島県徳島市川内町平石若松１０８番地４
(74)代理人　100105924
弁理士　森下　賢樹
(74)代理人　100109047
弁理士　村田　雄祐
(72)発明者　和家　伸明
徳島県徳島市ブレインズパーク　株式会社ジャストシステム内
(72)発明者　大島　教雄
徳島県徳島市ブレインズパーク　株式会社ジャストシステム内

最終頁に続く

(54)【発明の名称】複数のマークアップ表現における文書処理

(57)【要約】
【解決手段】文書処理システムは、自身が対応している第１のマークアップ言語により記述された文書を処理することが可能な処理系を備える。文書が、前記処理系が対応していない第２のマークアップ言語により記述されていたときに、前記文書を、前記第１のマークアップ言語にマッピングすることが可能な変換部が提供される。
【選択図】図１



【特許請求の範囲】
【請求項１】
自身が対応している第１のマークアップ言語により記述された文書を処理することが可能な処理系と、
文書が、前記処理系が対応していない第２のマークアップ言語により記述されていたときに、前記文書を、前記第１のマークアップ言語にマッピングすることが可能な変換部と、
を備えることを特徴とする文書処理装置。
【請求項２】
前記第１のマークアップ言語は、文書内のデータを、階層構造を有する複数の構成要素に分類して記述するための構造化言語であり、
前記変換部は、前記文書を、前記構成要素単位で前記第１のマークアップ言語にマッピングすることを特徴とする請求項１に記載の文書処理装置。
【請求項３】
前記第１及び第２のマークアップ言語の間のマッピングを記述した定義ファイルを更に備え、
前記変換部は、前記定義ファイルを参照して、前記文書を前記第１及び第２のマークアップ言語の間でマッピングすることを特徴とする請求項１に記載の文書処理装置。
【請求項４】
前記定義ファイルにおいて、前記構成要素の少なくとも一つは編集可能に定義されることを特徴とする請求項３に記載の文書処理装置。
【請求項５】
前記定義ファイルは、少なくとも一つの演算式を含み、
前記変換部は、前記演算式に基づいて演算を行った結果を、前記文書内の前記演算式に対応する位置に代入することを特徴とする請求項３に記載の文書処理装置。
【請求項６】
前記文書から、文書をデータとして扱うときのアクセス方法を提供するために定められた文書オブジェクトモデルに準拠した形式のデータを生成する生成部を更に備え、
前記生成部は、前記第２のマークアップ言語に対応する変換元文書オブジェクトモデルデータと、前記第１のマークアップ言語に対応する変換先文書オブジェクトモデルデータとを生成することを特徴とする請求項１に記載の文書処理装置。
【請求項７】
前記処理装置は、前記変換先文書オブジェクトモデルデータを参照して、前記マッピングされた文書を表示することを特徴とする請求項６に記載の文書処理装置。
【請求項８】
前記処理装置が、ユーザから前記文書の編集指示を受け付けたとき、
前記変換部は、前記変換元文書オブジェクトモデルデータと前記変換先文書オブジェクトモデルデータの該当箇所を変更し、
前記処理装置は、前記変換先文書オブジェクトモデルデータを参照して、表示を更新することを特徴とする請求項６に記載の文書処理装置。
【請求項９】
前記変換部は、前記変換元文書オブジェクトモデルデータに含まれるデータと、前記変換先文書オブジェクトモデルデータに含まれるデータとの対応関係を保持するコネクタ部を含むことを特徴とする請求項１に記載の文書処理装置。
【請求項１０】
前記処理装置が、ユーザから前記文書の編集指示を受け付けたとき、
前記コネクタ部は、前記編集指示を取得し、前記対応関係に基づいて、編集すべき前記変換元文書オブジェクトモデルデータのノードを抽出し、そのノードに対して、前記編集指示に対応した変更操作を実行することを特徴とする請求項９に記載の文書処理装置。
【請求項１１】

前記コネクタ部は、前記変換元文書オブジェクトモデルデータのノードが変更された旨の通知を受けたとき、前記対応関係に基づいて、変更されたノードに対応する前記変換先文書オブジェクトモデルデータのノードを抽出し、そのノードを再構築することを特徴とする請求項９に記載の文書処理装置。

【請求項１２】

前記コネクタ部は、前記変換先文書オブジェクトモデルデータの各ノードに１対１に対応したコネクタを含み、

前記コネクタは、前記変換先文書オブジェクトモデルデータのノードと前記変換元文書オブジェクトモデルデータのノードとの対応関係を保持する第１コネクタと、対応する前記変換元文書オブジェクトモデルデータのノードを有しない第２コネクタと、を含む

ことを特徴とする請求項９に記載の文書処理装置。

【請求項１３】

前記処理装置が、ユーザから前記文書の編集指示を受け付けたとき、

編集位置が属する前記変換先文書オブジェクトモデルデータのノードに対応する前記コネクタに前記編集指示が通知され、

前記編集指示が通知された前記コネクタが第１コネクタであった場合、その第１コネクタは、通知された編集指示を、対応する前記変換元文書オブジェクトモデルデータのノードに対する変更操作に変換することを特徴とする請求項１１に記載の文書処理装置。

【請求項１４】

前記コネクタ部は、前記変換元文書オブジェクトモデルデータのノードが変更された旨の通知を受けたとき、対応する前記変換先文書オブジェクトモデルデータのノードを再構築することを特徴とする請求項１２に記載の文書処理装置。

【請求項１５】

少なくとも１つの前記コネクタはテキストオブコネクタであり、前記テキストオブコネクタにより生成された対応ノードは編集可能であることを特徴とする請求項１２に記載の文書処理装置。

【請求項１６】

少なくとも１つの前記コネクタはバリューオブコネクタであり、前記バリューオブコネクタにより生成された対応ノードは編集不可能であることを特徴とする請求項１２に記載の文書処理装置。

【請求項１７】

少なくとも１つの前記コネクタはテンプレートコネクタであり、前記テンプレートコネクタに関連する対応ノードはテンプレートであり、前記テンプレートに関連する値は変更可能であることを特徴とする請求項１２に記載の文書処理装置。

【請求項１８】

少なくとも１つの前記コネクタは、前記変換元文書オブジェクトモデルデータの対応するノードのテキストの値が変更されても、前記ノードを特定可能であることを特徴とする請求項１２に記載の文書処理装置。

【請求項１９】

少なくとも１つの前記コネクタは、前記変換元文書オブジェクトモデルデータの最新の構造における最新のテキストを識別し、前記変換先文書オブジェクトモデルデータに通知可能であることを特徴とする請求項１８に記載の文書処理装置。

【請求項２０】

ｘｐａｔｈ表現を解決することにより、前記変換元文書オブジェクトモデルデータの対応するノードを特定することを特徴とする請求項１８に記載の文書処理装置。

【請求項２１】

前記ｘｐａｔｈ表現は、パスを含むことを特徴とする請求項２０に記載の文書処理装置。

【請求項２２】

前記ｘｐａｔｈ表現は、関数を含むことを特徴とする請求項２０に記載の文書処理装置

。
【請求項２３】
文書を処理する方法であって、
第１のマークアップ言語を処理不可能で第２のマークアップ言語を処理可能な文書処理装置により処理される前記文書が、前記第１のマークアップ言語で記述されていたときに、前記文書を前記第２のマークアップ言語にマッピングする過程と、
マッピングされた文書を表示する過程と、
を含むことを特徴とする文書処理方法。
【請求項２４】
第１のマークアップ言語を処理不可能で第２のマークアップ言語を処理可能な文書処理装置により処理される前記文書が、前記第１のマークアップ言語で記述されていたときに、前記文書を前記第２のマークアップ言語にマッピングする過程と、
マッピングされた文書を表示する過程と、
を含む手順を用いて文書処理操作をコンピュータに実行させることが可能な命令を有するコンピュータ読み取り可能な媒体を含むコンピュータプログラム製品。
【請求項２５】
文書処理装置により処理不可能な少なくとも１つのボキャブラリを含む文書を編集する方法であって、
文書をロードする過程と、
前記文書のソース文書オブジェクトモデルツリーを生成する過程と、
前記少なくとも１つのボキャブラリを処理するために適合された、前記文書のデスティネーション文書オブジェクトモデルツリーをツリー変換により生成する過程と、
を含むことを特徴とする方法。
【請求項２６】
編集操作を受け付けたとき、前記デスティネーション文書オブジェクトモデルツリーに変更を行う過程と、
前記ソース文書オブジェクトモデルツリーに対応する変更を行う過程と、
を更に含むことを特徴とする請求項２５に記載の方法。
【請求項２７】
前記ソース文書オブジェクトモデルツリーに対する前記変更は、前記ソース文書オブジェクトモデルツリーを修正するための一連のコマンドを生成することにより行われることを特徴とする請求項２６に記載の方法。
【請求項２８】
前記ソース文書オブジェクトモデルデータツリーと前記デスティネーション文書オブジェクトモデルデータツリーとの間の関係は、コネクタ部を用いて保持されることを特徴とする請求項２６に記載の方法。
【請求項２９】
前記コネクタ部は、複数のコネクタのツリーを含み、それぞれのコネクタは、前記デスティネーション文書オブジェクトモデルデータツリーにおける対応するノードを生成することを特徴とする請求項２８に記載の方法。
【請求項３０】
前記コネクタは、前記デスティネーション文書オブジェクトモデルデータのノードと前記ソース文書オブジェクトモデルデータのノードとの間の対応を保持する第１コネクタと、前記ソース文書オブジェクトモデルデータのノードを有しない第２コネクタとを含むことを特徴とする請求項２８に記載の方法。
【請求項３１】
少なくとも１つの前記コネクタはテキストオブコネクタであり、前記テキストオブコネクタにより生成された対応ノードは編集可能であることを特徴とする請求項２９に記載の方法。
【請求項３２】

少なくとも１つの前記コネクタはバリューオブコネクタであり、前記バリューオブコネクタにより生成された対応ノードは編集不可能であることを特徴とする請求項２９に記載の方法。
【請求項３３】
少なくとも１つの前記コネクタはテンプレートコネクタであり、前記テンプレートコネクタに関連する対応ノードはテンプレートであり、前記テンプレートに関連する値は変更可能であることを特徴とする請求項２９に記載の方法。
【請求項３４】
少なくとも１つの前記コネクタは、前記変換元文書オブジェクトモデルデータの対応するノードのテキストの値が変更されても、前記ノードを特定可能であることを特徴とする請求項２９に記載の方法。
【請求項３５】
少なくとも１つの前記コネクタは、前記変換元文書オブジェクトモデルデータの最新の構造における最新のテキストを識別し、前記変換先文書オブジェクトモデルデータに通知可能であることを特徴とする請求項３４に記載の方法。
【請求項３６】
ｘｐａｔｈ表現を解決することにより、前記変換元文書オブジェクトモデルデータの対応するノードを特定することを特徴とする請求項３４に記載の方法。
【請求項３７】
前記ｘｐａｔｈ表現は、パスを含むことを特徴とする請求項３５に記載の方法。
【請求項３８】
前記ｘｐａｔｈ表現は、関数を含むことを特徴とする請求項３５に記載の方法。
【請求項３９】
コマンドキューが生成されることを特徴とする請求項２７に記載の方法。
【請求項４０】
前記デスティネーション文書オブジェクトモデルデータツリーは、第２のソース文書オブジェクトモデルデータツリーとされ、ツリー変換を用いて第２のデスティネーション文書オブジェクトモデルデータツリーが生成されることを特徴とする請求項２５に記載の方法。
【請求項４１】
前記編集操作は、イベントとして実行されることを特徴とする請求項２６に記載の方法。
【請求項４２】
前記イベントは、マウスイベント又はキーボードイベントであることを特徴とする請求項４１に記載の方法。
【請求項４３】
前記ツリー変換は、定義ファイルに記述されることを特徴とする請求項２５に記載の方法。
【請求項４４】
前記定義ファイルは、イベントと少なくとも一つのコマンドとの間のマッピングを含むことを特徴とする請求項４３に記載の方法。
【請求項４５】
前記定義ファイルは、少なくとも一つの演算式を含み、前記演算式に基づいた計算結果が前記文書の前記演算式に対応する位置に代入されることを特徴とする請求項４３に記載の方法。
【請求項４６】
前記文書は、前記変換先文書オブジェクトモデルデータツリーに基づいて表示されることを特徴とする請求項２５に記載の方法。
【発明の詳細な説明】
【技術分野】

【０００１】
本発明は、文書処理技術に関し、特に、マークアップ言語により記述された文書を処理する技術に関する。
【背景技術】
【０００２】
インターネットの出現により、ユーザにより処理され管理される文書の数は、ほぼ指数関数的に増加してきた。インターネットの核を形成するＷＷＷ（World Wide Web：「ウェブ」とも呼ばれる）は、そのような文書の大きなデータ蓄積領域を含む。ウェブは、文書だけでなく、文書のための情報検索システムを提供する。これらの文書の多くはマークアップ言語で整形されている。シンプルかつ普及しているマークアップ言語の一つにＨＴＭＬ（HyperText Markup Language）がある。このような文書は、ウェブの他の部分にある別の文書へのリンクも含んでいる。ＸＭＬ（eXtensible Markup Language）は、更に高度で普及しているマークアップ言語である。ウェブを介して文書にアクセスし閲覧するためのシンプルなブラウザが、Ｊａｖａ（登録商標）などのオブジェクト指向プログラム言語により開発されている。
【０００３】
マークアップ言語により整形された文書は、通常、ブラウザや他のアプリケーションの中では、ツリーデータ構造の形で表現される。この構造は、文書のパースツリーに相当する。文書オブジェクトモデル（ＤＯＭ：Document Object Model）は、文書を表現し、操作するために使用される、よく知られたツリーベースのデータ構造モデルである。ＤＯＭは、ＨＴＭＬやＸＭＬ文書を含む文書を表現するための標準的なオブジェクトのセットを提供する。ＤＯＭは、文書内の要素を表現するオブジェクトがどのように結合されるかという標準的なモデルと、それらのオブジェクトにアクセスし操作するための標準的なインタフェイスという、２つの基本的なコンポーネントを含む。
【０００４】
アプリケーション開発者は、独自のデータ構造やＡＰＩ（Application Program Interface）へのインタフェイスとしてＤＯＭをサポートすることができる。他方、文書を作成するアプリケーション開発者は、自身のＡＰＩの独自インタフェイスではなく、ＤＯＭの標準インタフェイスを使用することができる。したがって、標準を提供するというその能力により、ＤＯＭは、様々な環境、特にウェブにおいて、文書の相互利用を増加させるために有効である。ＤＯＭのいくつかのバージョンが定義されており、異なるプログラミング環境及びアプリケーションによって使用されている。
【０００５】
ＤＯＭツリーは、対応するＤＯＭの内容に基づいた文書の階層的表現である。ＤＯＭツリーは「根（ルート）」を含み、ルートから生じる１以上の「節（ノード）」を含む。ルートが文書全体を表現することもある。中間のノードは、例えば、テーブルや、テーブルの行及び列などの要素を表現することもできる。ＤＯＭツリーの「葉」は、通常、テキスト項目やイメージなど、それ以上分解できないデータを表現する。ＤＯＭツリーのそれぞれのノードには、ノードにより表現される要素のパラメータ、例えばフォント、サイズ、色、インデントなどを記述する属性を関連づけることができる。
【０００６】
ＨＴＭＬは、文書を作成するために一般に用いられる言語であるが、フォーマット及びレイアウト用の言語であり、データ記述のための言語ではない。ＨＴＭＬドキュメントを表現するＤＯＭツリーのノードは、ＨＴＭＬの整形タグとして予め定義された要素であって、通常、ＨＴＭＬは、データの詳述や、データのタギング／ラベリングのための機能を提供しないので、ＨＴＭＬドキュメント中のデータに対するクエリを定式化することは多くの場合困難である。
【０００７】
ネットワーク設計者たちの目指すものは、ウェブ上の文書がソフトウェアアプリケーションによってクエリされたり処理されたりできるようにすることである。表示方法とは無

関係で、階層的に構造化された言語であれば、そのようにクエリされ処理されることができる。ＸＭＬ（eXtensible Markup Language）のようなマークアップ言語は、これらの特徴を提供することができる。

【０００８】

ＨＴＭＬとは逆に、ＸＭＬのよく知られた利点は、自由に定義可能な「タグ」を用いて、文書の設計者がデータ要素にラベルを付すことができることである。このようなデータ要素は階層的に体系化することができる。さらに、ＸＭＬ文書は、文書において用いられる「文法」（タグ及びタグ間の相互関係）を記述した文書型定義（ＤＴＤ：Document Type Definition）を含むことができる。構造化されたＸＭＬ文書の表示方法を定義するために、ＣＳＳ（Cascading Style Sheets）又はＸＳＬ（XML Style Language）が用いられる。ＤＯＭ、ＨＴＭＬ、ＸＭＬ、ＣＳＳ、ＸＳＬ及び関連する言語の特徴に関する更なる情報は、ウェブ、例えば、「http://www.w3.org/TR/」から入手可能である。

【０００９】

Ｘｐａｔｈは、ＸＭＬ文書の部分の位置を指定するための共通のシンタックス及びセマンティクスである。Ｘｐａｔｈの機能として、例えばＸＭＬ文書に対応するＤＯＭツリーのトラバースがある。これにより、ＸＭＬ文書の様々な表現に関連づけられた文、数値、及びブーリアンキャラクタを操作するための基本的な機能が提供される。Ｘｐａｔｈは、ＸＭＬ文書の見た目のシンタックス、例えば、テキストとしてみたときに何行目であるとか何文字目であるとかといった文法ではなく、抽象的、論理的な構造、例えばＤＯＭツリー上で動作する。Ｘｐａｔｈを使用することにより、例えばＸＭＬ文書のＤＯＭツリー内の階層的構造を通じて場所を指定することができる。位置を指定する以外にも、Ｘｐａｔｈは、ＤＯＭツリー中のノードがパターンにマッチするか否かを判定するために使用可能に設計されている。

【００１０】

Ｘｐａｔｈに関する更なる詳細は、「http://www.w3.org/TR/xpath」で入手可能である。

【００１１】

ＸＭＬについて既知の利点及び特徴を鑑みると、ＸＭＬなどのマークアップ言語による文書を扱うことが可能で、文書を作成し編集するためのユーザフレンドリーなインタフェースを提供する効果的な文書処理管理システムが必要である。ＸＭＬは、複合文書や、ネットワークなどを介して他者とデータを共有するのに適した形式として注目されており、ＸＭＬ文書を作成、表示、編集するためのアプリケーションが開発されている（たとえば、特許文献１参照）。ＸＭＬ文書は、文書型定義などにより定義されたボキャブラリ（タグセット）に基づいて作成されている。

【特許文献１】特開２００１－２９０８０４号公報

【発明の開示】

【発明が解決しようとする課題】

【００１２】

ボキャブラリは、任意に定義することが許されており、理論上、無限に多くのボキャブラリが存在しうる。これらのボキャブラリの全てに対応して専用の表示・編集環境を提供するのは現実的ではない。従来、専用の編集環境が用意されていないボキャブラリにより記述された文書を編集する場合、テキストデータにより構成された文書のソースを直接テキストエディタなどで編集していた。

【００１３】

ＸＭＬ文書を扱うことが可能な既存のアプリケーションは市場で入手可能であるが、広く受け入れられることを阻害する重大な限界や障壁を有している。本明細書で説明する方法及び装置は、このような既存の製品やその基礎となる既存技術によりこれまで取り組まれてこなかった問題を解決する。

【００１４】

例えば、既存のＸＭＬ文書処理装置の実装において、表示の方法に関連しない内容の表

現というＸＭＬ文書の特徴は、表面上は利点であるとみなされている。しかし、このような特徴は現実的にはユーザが直接編集することができないという不都合を生じる。この問題を解決するために、既存のＸＭＬ文書処理製品は、ＸＭＬ入力のための固有の画面を設計している。しかし、既存のＸＭＬ製品は予めハードコードされたものであるので、画面設計の柔軟性は制限される。

【００１５】

この制限に関して、スタイルシート言語の標準の一つとしてＸＳＬＴが既に開発されている。これは、ユーザをハードコードから解放する技術であり、ＸＭＬ文書の表示に適用可能な方法と両立可能である。しかし、ＸＳＬＴはＸＭＬ文書を表示するだけであり、編集することを可能にするわけではない。

【００１６】

さらに、既存のＸＭＬ製品は、主として「スキーマ」の存在に依存している。したがって、最初にいったんスキーマが決定されると、最上層からスキーマ構造に対応したＸＭＬ文書のみしか扱うことができないという制限がある。言い換えれば、システムは固定したシステムである。

【課題を解決するための手段】

【００１７】

本発明によれば、上述した制限はない。ＸＭＬ文書全体の構造が固定的に決定される必要はない。様々な構造を持つ複合ＸＭＬ文書は、ＸＭＬ文書をいくつかの部分に分割し、編集モジュールを割り当てるというアイデアにより、安全に取り扱うことができる。柔軟なシステムを実現するために、編集モジュールはプラグインにより実現されることが好ましい。さらに、ユーザは、ハードコードの制限なしに、柔軟な画面設計を実装し、ＷＹＳＩＷＹＧにより編集することができる。

【００１８】

本発明はこうした状況に鑑みてなされたものであり、その目的は、１以上のＸＭＬなどのマークアップ言語により記述された文書を効果的に処理する装置、方法、プログラム製品を提供することにある。

【００１９】

本発明のある態様は、文書処理装置に関する。この文書処理装置は、自身が対応している第１のマークアップ言語により記述された文書を処理することが可能な処理系と、文書が、前記処理系が対応していない第２のマークアップ言語により記述されていたときに、前記文書を、前記第１のマークアップ言語にマッピングすることが可能な変換部と、を備える。

【００２０】

本発明の別の態様は、文書処理方法に関する。この文書処理方法は、第１のマークアップ言語を処理不可能で第２のマークアップ言語を処理可能な文書処理装置により処理される前記文書が、前記第１のマークアップ言語で記述されていたときに、前記文書を前記第２のマークアップ言語にマッピングする過程と、マッピングされた文書を表示する過程と、を備える。

【００２１】

上述した技術をコンピュータに実行させることが可能な命令を有するコンピュータ読み取り可能な媒体を含むコンピュータプログラム製品も、本発明の一態様である。

【００２２】

なお、以上の構成要素の任意の組合せ、本発明の表現を方法、装置、システムなどの間で変換したものもまた、本発明の態様として有効である。

【発明の効果】

【００２３】

本発明によれば、１以上のマークアップ言語により記述された文書を、生成、編集、表示、又は保存のうち少なくとも１つのために適切に処理する技術を提供することができる。

【発明を実施するための最良の形態】
【００２４】
図１は、実施の形態に係る文書処理装置２０の構成を示す。文書処理装置２０は、文書内のデータが階層構造を有する複数の構成要素に分類された構造化文書を処理するが、本実施の形態では構造化文書の一例としてＸＭＬ文書を処理する例について説明する。文書処理装置２０は、主制御ユニット２２、編集ユニット２４、ＤＯＭ（Document Object Model）ユニット３０、ＣＳＳ（Cascade Style Sheets）ユニット４０、ＨＴＭＬ（HyperText Markup Language）ユニット５０、ＳＶＧ（Scalable Vector Graphics）ユニット６０、及び変換部の一例であるＶＣ（Vocabulary Connection）ユニット８０を備える。これらの構成は、ハードウエアコンポーネントでいえば、任意のコンピュータのＣＰＵ、メモリ、メモリにロードされたプログラムなどによって実現されるが、ここではそれらの連携によって実現される機能ブロックを描いている。したがって、これらの機能ブロックがハードウエアのみ、ソフトウエアのみ、またはそれらの組合せによっていろいろな形で実現できることは、当業者には理解されるところである。
【００２５】
主制御ユニット２２は、プラグインのロードや、コマンド実行のフレームワークを提供する。編集ユニット２４は、ＸＭＬ文書を編集するためのフレームワークを提供する。文書処理装置２０における文書の表示及び編集機能は、プラグインにより実現されており、文書の種別に応じて必要なプラグインが主制御ユニット２２又は編集ユニット２４によりロードされる。主制御ユニット２２又は編集ユニット２４は、処理対象となるＸＭＬ文書の名前空間を参照して、ＸＭＬ文書がいずれのボキャブラリにより記述されているかを判別し、そのボキャブラリに対応した表示又は編集用のプラグインをロードして表示や編集を実行させる。例えば、文書処理装置２０には、制御部５２、編集部５４、及び表示部５６を用いてＨＴＭＬ文書の表示及び編集を行うＨＴＭＬユニット５０や、制御部６２、編集部６４、及び表示部６６を用いてＳＶＧ文書の表示及び編集を行うＳＶＧユニット６０など、ボキャブラリ（タグセット）ごとに表示系及び編集系がプラグインとして実装されており、ＨＴＭＬ文書を編集するときはＨＴＭＬユニット５０が、ＳＶＧ文書を編集するときはＳＶＧユニット６０が、それぞれの制御部と協働してロードされる。後述するように、ＨＴＭＬとＳＶＧの双方の構成要素を含む複合文書が処理対象となっている場合は、ＨＴＭＬユニット５０とＳＶＧユニット６０の双方がロードされる。
【００２６】
このような構成によれば、ユーザは、必要な機能のみを選択してインストールし、後から適宜機能を追加又は削除することができるので、プログラムを格納するハードディスクなどの記録媒体の記憶領域を有効に活用することができ、また、プログラム実行時にも、メモリの浪費を防ぐことができる。また、機能拡張性に優れており、開発主体としても、プラグインの形で新たなボキャブラリに対応することが可能なので開発が容易となり、ユーザとしても、プラグインの追加により容易かつ低コストにて機能を追加することができる。
【００２７】
編集ユニット２４は、ユーザインターフェースを介してユーザからマウスクリックやキーストロークなどの入力操作を含む編集指示のイベント（トリガイベント）を受け付け、そのイベントを適切なプラグインなどに通知するともに、イベントの再実行（リドゥ）又は実行の取消（アンドゥ）などの処理を制御する。
【００２８】
ＤＯＭユニット３０は、ＤＯＭ提供部３２、ＤＯＭ生成部３４、及び出力部３６を含み、ＸＭＬ文書をデータとして扱うときのアクセス方法を提供するために定められた文書オブジェクトモデル（Document Object Model：ＤＯＭ）に準拠した機能を実現する。ＤＯＭ提供部３２は、編集ユニット２４に定義されているインタフェースを満たすＤＯＭの実装である。ＤＯＭ生成部３４は、ＸＭＬ文書からＤＯＭツリーを生成する。後述するように、処理対象となるＸＭＬ文書が、ＶＣユニット８０により他のボキャブラリにマッピン

グされる場合は、マッピング元のＸＭＬ文書に対応するソースツリーと、マッピング先のＸＭＬ文書に対応するデスティネーションツリーが生成される。出力部３６は、例えば編集終了時に、ＤＯＭツリーをＸＭＬ文書として出力する。

【００２９】

ＣＳＳユニット４０は、ＣＳＳ解析部４２、ＣＳＳ提供部４４、及びレンダリング部４６を含み、ＣＳＳに準拠した表示機能を提供する。ＣＳＳ解析部４２は、ＣＳＳの構文を解析するパーサの機能を有する。ＣＳＳ提供部４４は、ＣＳＳオブジェクトの実装であり、ＤＯＭツリーに対してＣＳＳのカスケード処理を行う。レンダリング部４６は、ＣＳＳのレンダリングエンジンであり、ＣＳＳを用いてレイアウトされるＨＴＭＬなどのボキャブラリで記述された文書の表示に用いられる。

【００３０】

ＨＴＭＬユニット５０は、ＨＴＭＬにより記述された文書を表示又は編集する。ＳＶＧユニット６０は、ＳＶＧにより記述された文書を表示又は編集する。これらの表示／編集系は、プラグインの形で実現されており、それぞれ、文書を表示する表示部（Canvas）５６、６６、編集指示を含むイベントを送受信する制御部（Editlet）５２、６２、編集コマンドを受けてＤＯＭに対して編集を行う編集部（Zone）５４、６４を備える。制御部５２又は６２が外部からＤＯＭツリーの編集コマンドを受け付けると、編集部５４又は６４がＤＯＭツリーを変更し、表示部５６又は６６が表示を更新する。これらは、よく知られたグラフィカルユーザインタフェース（ＧＵＩ）のパラダイムである、ＭＶＣ（Model-View-Controller）と呼ばれるフレームワークに類似する構成をとっている。ＭＶＣパラダイムは、アプリケーション又はアプリケーションのインタフェイスの一部を、３つの部分、すなわち、モデル、ビュー、コントローラに分割する方法を提案する。ＭＶＣは、元は、ＧＵＩの世界に、従来の入力、処理、出力の役割を割り当てるために開発された。

［入力］ → ［処理］ → ［出力］

［コントローラ］→ ［モデル］ → ［ビュー］

【００３１】

ＭＶＣパラダイムによれば、外界のモデリング、ユーザへの視覚的なフィードバック、及びユーザの入力は、モデル（Ｍ）、ビュー（Ｖ）、及びコントローラ（Ｃ）オブジェクトにより分離されて扱われる。コントローラは、ユーザからのマウスとキーボード入力のような入力を解釈し、これらのユーザアクションを、適切な変更をもたらすためにモデル及び／又はビューに送られるコマンドにマップするように作用する。モデルは、１以上のデータ要素を管理するように作用し、その状態に関するクエリに応答し、状態を変更する指示に応答する。ビューは、ディスプレイの長方形の領域を管理するように作用し、グラフィクスとテキストの組合せによりユーザにデータを提示する機能を有する。

【００３２】

ここで開示される本発明の実施の形態では、概ね、表示部５６及び６６が「View」に、制御部５２及び６２が「Controller」に、編集部５４及び６４とＤＯＭの実体が「Model」に、それぞれ対応する。図１－１０に示す本実施の形態の文書処理装置２０では、ＸＭＬ文書をツリー表示形式で編集するだけでなく、それぞれのボキャブラリに応じた編集を可能とする。例えば、ＨＴＭＬユニット５０は、ＨＴＭＬ文書をワードプロセッサに類似した方式で編集するためのユーザインターフェースを提供し、ＳＶＧユニット６０は、ＳＶＧ文書を画像描画ツールに類似した方式で編集するためのユーザインターフェースを提供する。

【００３３】

ＶＣユニット８０は、マッピング部８２、定義ファイル取得部８４、及び定義ファイル生成部８６を含み、あるボキャブラリにより記述された文書を、他のボキャブラリにマッピングすることにより、マッピング先のボキャブラリに対応した表示編集用プラグインで文書を表示又は編集するためのフレームワークを提供する。本実施の形態では、この機能を、ボキャブラリコネクション（Vocabulary Connection：ＶＣ）と呼ぶ。定義ファイル取得部８４は、マッピングの定義を記述した定義ファイルを取得する。本実施の形態では

、定義ファイルはスクリプトファイルである。

【００３４】

第１のボキャブラリの文書は、ノードを含むソースツリーとして表現される。同様に、第２のボキャブラリの文書は、ノードを含むデスティネーションツリーとして表現される。定義ファイルは、ノードごとに、ソースツリーとデスティネーションツリーのノード間の対応（コネクション）を記述する。Ｗ３Ｃの技術において知られているように、ＤＯＭツリー中のノードは、要素値及び／又は属性値により定義されてもよい。本実施の形態では、各ノードの要素値や属性値の編集の可否を指定してもよい。

【００３５】

また、ノードの要素値や属性値を用いた演算式を記述してもよい。これらの機能については、後で詳述する。マッピング部８２は、定義ファイル取得部８４が取得した定義ファイル（スクリプトファイル）を参照して、ＤＯＭ生成部３４にデスティネーションツリーを生成させ、ソースツリーとデスティネーションツリーの対応関係を管理する。定義ファイル生成部８６は、ユーザが定義ファイルを生成するためのグラフィカルユーザインターフェースを提供する。

【００３６】

ＶＣユニット８０は、ソースツリーとデスティネーションツリーの間のコネクションを監視し、表示を担当するプラグインにより提供されるユーザインタフェースを介してユーザから編集指示を受け付けると、まずソースツリーの該当するノードを変更する。ＤＯＭユニット３０が、ソースツリーが変更された旨のミューテーションイベントを発行すると、ＶＣユニット８０は、そのミューテーションイベントを受けて、ソースツリーの変更にデスティネーションツリーを同期させるべく、変更されたノードに対応するデスティネーションツリーのノードを変更する。デスティネーションツリーを表示／編集するプラグイン、例えばＨＴＭＬユニット５０は、デスティネーションツリーが変更された旨のミューテーションイベントを受けて、変更されたデスティネーションツリーを参照して表示を更新する。このような構成により、少数のユーザにより利用されるローカルなボキャブラリにより記述された文書であっても、他のメジャーなボキャブラリに変換することで、文書を表示することができるとともに、編集環境が提供される。

【００３７】

文書処理装置２０により文書を表示又は編集する動作について説明する。文書処理装置２０が処理対象となる文書を読み込むと、ＤＯＭ生成部３４が、そのＸＭＬ文書からＤＯＭツリーを生成する。また、主制御ユニット２２又は編集ユニット２４は、名前空間を参照して文書を記述しているボキャブラリを判別する。そのボキャブラリに対応したプラグインが文書処理装置２０にインストールされている場合は、そのプラグインをロードして、文書を表示／編集させる。プラグインがインストールされていない場合は、マッピングの定義ファイルが存在するか否かを確認する。定義ファイルが存在する場合、定義ファイル取得部８４が定義ファイルを取得し、その定義に従って、デスティネーションツリーが生成され、マッピング先のボキャブラリに対応するプラグインにより文書が表示／編集される。複数のボキャブラリを含む複合文書である場合は、後述するように、それぞれのボキャブラリに対応したプラグインにより、文書の該当箇所がそれぞれ表示／編集される。定義ファイルが存在しない場合は、文書のソース又はツリー構造を表示し、その表示画面において編集が行われる。

【００３８】

図２は、処理対象となるＸＭＬ文書の例を示す。このＸＭＬ文書は、生徒の成績データを管理するために用いられる。ＸＭＬ文書のトップノードである構成要素「成績」は、配下に、生徒ごとに設けられた構成要素「生徒」を複数有する。構成要素「生徒」は、属性値「名前」と、子要素「国語」、「数学」、「理科」、「社会」を有する。属性値「名前」は、生徒の名前を格納する。構成要素「国語」、「数学」、「理科」、「社会」は、それぞれ、国語、数学、理科、社会の成績を格納する。例えば、名前が「Ａ」である生徒の国語の成績は「９０」、数学の成績は「５０」、理科の成績は「７５」、社会の成績は「

６０」である。以下、この文書で使用されているボキャブラリ（タグセット）を、「成績管理ボキャブラリ」と呼ぶ。

【００３９】

本実施の形態の文書処理装置２０は、成績管理ボキャブラリの表示／編集に対応したプラグインを有しないので、この文書をソース表示、ツリー表示以外の方法で表示するためには、前述したＶＣ機能が用いられる。すなわち、成績管理ボキャブラリを、プラグインが用意された別のボキャブラリ、例えば、ＨＴＭＬやＳＶＧなどにマッピングするための定義ファイルを用意する必要がある。ユーザ自身が定義ファイルを作成するためのユーザインターフェースについては後述することにして、ここでは、既に定義ファイルが用意されているとして説明を進める。

【００４０】

図３は、図２に示したＸＭＬ文書をＨＴＭＬで記述された表にマッピングする例を示す。図３の例では、成績管理ボキャブラリの「生徒」ノードを、ＨＴＭＬにおける表（「TABLE」ノード）の行（「TR」ノード）に対応づけ、各行の第１列には属性値「名前」を、第２列には「国語」ノードの要素値を、第３列には「数学」ノードの要素値を、第４列には「理科」ノードの要素値を、第５列には「社会」ノードの要素値を、それぞれ対応付ける。これにより、図２に示したＸＭＬ文書を、ＨＴＭＬの表形式で表示することができる。また、これらの属性値及び要素値は、編集可能であることが指定されており、ユーザがＨＴＭＬによる表示画面上で、ＨＴＭＬユニット５０の編集機能により、これらの値を編集することができる。第６列には、国語、数学、理科、社会の成績の加重平均を算出する演算式が指定されており、生徒の成績の平均点が表示される。このように、定義ファイルに演算式を指定可能とすることにより、より柔軟な表示が可能となり、編集時のユーザの利便性を向上させることができる。なお、第６列は、編集不可であることが指定されており、平均点のみを個別に編集することができないようにしている。このように、マッピング定義において、編集の可否を指定可能とすることにより、ユーザの誤操作を防ぐことができる。

【００４１】

図４は、図２に示したＸＭＬ文書を図３に示した表にマッピングするための定義ファイルの例を示す。この定義ファイルは、定義ファイル用に定義されたスクリプト言語により記述される。定義ファイルには、コマンドの定義と、表示のテンプレートが記述されている。図４の例では、コマンドとして、「生徒の追加」と「生徒の削除」が定義されており、それぞれ、ソースツリーにノード「生徒」を挿入する操作と、ソースツリーからノード「生徒」を削除する操作が対応付けられている。また、テンプレートとして、表の第１行に「名前」、「国語」などの見出しが表示され、第２行以降に、ノード「生徒」の内容が表示されることが記述されている。ノード「生徒」の内容を表示するテンプレート中、「text-of」と記述された項は「編集可能」であることを意味し、「value-of」と記述された項は「編集不可能」であることを意味する。また、ノード「生徒」の内容を表示する行のうち、第６列には、「(src:国語 + src:数学 + src:理科 + src:社会) div 4」という計算式が記述されており、生徒の成績の平均が表示されることを意味する。

【００４２】

図５は、図２に示した成績管理ボキャブラリで記述されたＸＭＬ文書を、図３に示した対応によりＨＴＭＬにマッピングして表示した画面の例を示す。表９０の各行には、左から、各生徒の名前、国語の成績、数学の成績、理科の成績、社会の成績、及び平均点が表示されている。ユーザは、この画面上で、ＸＭＬ文書を編集することができる。たとえば、第２行第３列の値を「７０」に変更すると、このノードに対応するソースツリーの要素値、すなわち、生徒「Ｂ」の数学の成績が「７０」に変更される。このとき、ＶＣユニット８０は、デスティネーションツリーをソースツリーに追従させるべく、デスティネーションツリーの該当箇所を変更し、ＨＴＭＬユニット５０が、変更されたデスティネーションツリーに基づいて表示を更新する。したがって、生徒「Ｂ」の数学の成績が「７０」に変更され、更に、平均点が「５５」に変更される。

【００４３】
図５に示した画面には、図４に示した定義ファイルに定義されたように、「生徒の追加」及び「生徒の削除」のコマンドがメニューに表示される。ユーザがこれらのコマンドを選択すると、ソースツリーにおいて、ノード「生徒」が追加又は削除される。このように、本実施の形態の文書処理装置２０では、階層構造の末端の構成要素の要素値を編集するのみではなく、階層構造を編集することも可能である。このようなツリー構造の編集機能は、コマンドの形でユーザに提供されてもよい。また、例えば、表の行を追加又は削除するコマンドが、ノード「生徒」を追加又は削除する操作に対応づけられてもよい。また、他のボキャブラリを埋め込むコマンドがユーザに提供されてもよい。この表を入力用テンプレートとして、穴埋め形式で新たな生徒の成績データを追加することもできる。以上のように、ＶＣ機能により、ＨＴＭＬユニット５０の表示／編集機能を利用しつつ、成績管理ボキャブラリで記述された文書を編集することが可能となる。

【００４４】
図６は、ユーザが定義ファイルを生成するために、定義ファイル生成部８６がユーザに提示するグラフィカルユーザインタフェースの例を示す。画面左側の領域９１には、マッピング元のＸＭＬ文書がツリー表示されている。画面右側の領域９２には、マッピング先のＸＭＬ文書の画面レイアウトが示されている。この画面レイアウトは、ＨＴＭＬユニット５０により編集可能となっており、ユーザは、画面右側の領域９２において、文書を表示するための画面レイアウトを作成する。そして、例えば、マウスなどのポインティングデバイスにより、画面左側の領域９１に表示されたマッピング元のＸＭＬ文書のノードを、画面右側の領域９２に表示されたＨＴＭＬによる画面レイアウト中へドラッグ＆ドロップ操作を行うことにより、マッピング元のノードと、マッピング先のノードとのコネクションが指定される。例えば、要素「生徒」の子要素である「数学」を、ＨＴＭＬ画面の表９０の第１行第３列にドロップすると、「数学」ノードと、３列目の「ＴＤ」ノードの間にコネクションが張られる。各ノードには、編集の可否が指定できるようになっている。また、表示画面中には、演算式を埋め込むこともできる。画面の編集が終わると、定義ファイル生成部８６は、画面レイアウトとノード間のコネクションを記述した定義ファイルを生成する。

【００４５】
ＸＨＴＭＬ、ＭａｔｈＭＬ、ＳＶＧなどの主要なボキャブラリに対応したビューワやエディタは既に開発されているが、図２に示した文書のようなオリジナルなボキャブラリで記述された文書に対応したビューワやエディタを開発するのは現実的でない。しかし、上記のように、他のボキャブラリにマッピングするための定義ファイルを作成すれば、ビューワやエディタを開発しなくても、ＶＣ機能を利用して、オリジナルなボキャブラリで記述された文書を表示・編集することができる。

【００４６】
図７は、定義ファイル生成部８６により生成された画面レイアウトの他の例を示す。図７の例では、成績管理ボキャブラリで記述されたＸＭＬ文書を表示するための画面に、表９０と、円グラフ９３が作成されている。この円グラフ９３は、ＳＶＧにより記述される。後述するように、本実施の形態の文書処理装置２０は、一つのＸＭＬ文書内に複数のボキャブラリを含む複合文書を処理することができるので、この例のように、ＨＴＭＬで記述された表９０と、ＳＶＧで記述された円グラフ９３とを、一つの画面上に表示することができる。

【００４７】
図８は、文書処理装置２０によるＸＭＬ文書の編集画面の一例を示す。図８の例では、一つの画面が複数に分割されており、それぞれの領域において、処理対象となるＸＭＬ文書を異なる複数の表示形式により表示している。領域９４には、文書のソースが表示されており、領域９５には、文書のツリー構造が表示されており、領域９６には、図５に示したＨＴＭＬにより記述された表が表示されている。これらのいずれの画面上においても、文書の編集が可能であり、いずれかの画面上でユーザが編集を行うと、ソースツリーが変

更され、それぞれの画面の表示を担当するプラグインが、ソースツリーの変更を反映すべく画面を更新する。具体的には、ソースツリーの変更を通知するミューテーションイベントのリスナーとして、それぞれの編集画面の表示を担当するプラグインの表示部を登録しておき、いずれかのプラグイン又はＶＣユニット８０によりソースツリーが変更されたときに、編集画面を表示中の全ての表示部が、発行されたミューテーションイベントを受け取って画面を更新する。このとき、プラグインがＶＣ機能により表示を行っている場合は、ＶＣユニット８０がソースツリーの変更に追従してデスティネーションツリーを変更した後、変更されたデスティネーションツリーを参照してプラグインの表示部が画面を更新する。

【００４８】

例えば、ソース表示及びツリー表示を、専用のプラグインにより実現している場合は、ソース表示用プラグインとツリー表示用プラグインは、デスティネーションツリーを用いず、直接ソースツリーを参照して表示を行う。この場合、いずれかの画面において編集が行われると、ソース表示用プラグインとツリー表示用プラグインは、変更されたソースツリーを参照して画面を更新し、領域９６の画面を担当しているＨＴＭＬユニット５０は、ソースツリーの変更に追従して変更されたデスティネーションツリーを参照して画面を更新する。

【００４９】

ソース表示及びツリー表示は、ＶＣ機能を利用して実現することもできる。すなわち、ソース、ツリー構造をＨＴＭＬによりレイアウトし、そのＨＴＭＬにＸＭＬ文書をマッピングして、ＨＴＭＬユニット５０により表示してもよい。この場合、ソース形式、ツリー形式、表形式の３つのデスティネーションツリーが生成されることになる。いずれかの画面において編集が行われると、ＶＣユニット８０は、ソースツリーを変更した後、ソース形式、ツリー形式、表形式の３つのデスティネーションツリーをそれぞれ変更し、ＨＴＭＬユニット５０は、それらのデスティネーションツリーを参照して、３つの画面を更新する。

【００５０】

このように、一つの画面上に複数の表示形式で文書を表示することにより、ユーザの利便性を向上させることができる。例えば、ユーザは、ソース表示又はツリー表示により文書の階層構造を把握しつつ、表９０などを用いて視覚的に分かりやすい形式で文書を表示し、編集することができる。上記の例では、一つの画面を分割して複数の表示形式による画面を同時に表示したが、一つの画面に一つの表示形式による画面を表示し、表示形式をユーザの指示により切り替え可能としてもよい。この場合、主制御ユニット２２が、ユーザから表示形式の切り替え要求を受け付け、各プラグインに指示して表示を切り替える。

【００５１】

図９は、文書処理装置２０により編集されるＸＭＬ文書の他の例を示す。図９に示したＸＭＬ文書では、ＳＶＧ文書の「foreignObject」タグの中にＸＨＴＭＬ文書が埋め込まれており、さらに、ＸＨＴＭＬ文書の中にＭａｔｈＭＬで記述された数式が入っている。このような場合、編集ユニット２４が、名前空間を参照して、適切な表示系に描画作業を振り分ける。図９の例では、編集ユニット２４は、まず、ＳＶＧユニット６０に四角形を描画させ、つづいて、ＨＴＭＬユニット５０にＸＨＴＭＬ文書を描画させる。さらに、図示しないＭａｔｈＭＬユニットに、数式を描画させる。こうして、複数のボキャブラリを包含する複合文書が適切に表示される。表示結果を図１０に示す。

【００５２】

文書編集中、編集メニューがユーザに表示されてもよい。メニューは編集対象の複合文書の位置に対応してもよい。このように、ユーザにより表示画面上を移動されるカーソル（キャリッジ）の位置に応じて、表示されるメニューを切り替えてもよい。すなわち、カーソルが、ＳＶＧ文書が表示された領域内に存在するときは、ＳＶＧユニット６０が提供するメニュー、又はＳＶＧ文書をマッピングするための定義ファイルに定義されたコマンドを表示し、カーソルが、ＸＨＴＭＬ文書が表示された領域内に存在するときは、ＨＴＭ

Ｌユニット５０が提供するメニュー、又はＸＨＴＭＬ文書をマッピングするための定義ファイルに定義されたコマンドを表示する。これにより、編集位置に応じて適切なユーザインターフェースを提供することができる。

【００５３】

複合文書において、あるボキャブラリに対応する適切なプラグイン又はマッピング定義ファイルがなかった場合は、そのボキャブラリにより記述された部分は、ソース表示又はツリー表示されてもよい。従来、ある文書に他の文書を埋め込んだ複合文書を開くとき、埋め込まれた文書を表示するアプリケーションがインストールされていないと、その内容を表示することができなかったが、本実施の形態では、表示用のアプリケーションが存在しなくても、テキストデータにより構成されたＸＭＬ文書をソース表示又はツリー表示することにより内容を把握することができる。これは、テキストベースであるＸＭＬなどの文書ならではの特徴といえる。

【００５４】

データがテキストベースで記述されることの他の利点として、例えば、複合文書中の、あるボキャブラリにより記述される部分において、同一文書内の他のボキャブラリで記述された部分のデータを参照してもよい。また、文書内で検索を実行する時に、ＳＶＧなどの図に埋め込まれた文字列も検索対象とすることができる。

【００５５】

あるボキャブラリにより記述された文書内に、他のボキャブラリのタグを用いてもよい。このＸＭＬ文書は、妥当（valid）ではないが、整形式（well-formed）であれば、有効なＸＭＬ文書として処理可能である。この場合、挿入された他のボキャブラリのタグは、定義ファイルによりマッピングされてもよい。例えば、ＸＨＴＭＬ文書中に、「重要」、「最重要」などのタグを使用し、これらのタグで囲まれた部分を強調表示してもよいし、重要度の順にソートして表示してもよい。

【００５６】

図１０に示した編集画面において、ユーザにより文書が編集されると、編集された部分を担当するプラグイン又はＶＣユニット８０がソースツリーを変更する。ソースツリーには、ノードごとにミューテーションイベントのリスナーを登録できるようになっており、通常は、各ノードが属するボキャブラリに対応したプラグインの表示部又はＶＣユニット８０がリスナーとして登録される。ＤＯＭ提供部３２は、ソースツリーが変更されると、変更されたノードから上位の階層へたどって、登録されたリスナーがあれば、そのリスナーへミューテーションイベントを発行する。例えば、図９に示した文書において、＜ｈｔｍｌ＞ノードの下位のノードが変更された場合、＜ｈｔｍｌ＞ノードにリスナーとして登録されたＨＴＭＬユニット５０にミューテーションイベントが通知されるとともに、その上位の＜ｓｖｇ＞ノードにリスナーとして登録されたＳＶＧユニット６０にもミューテーションイベントが通知される。このとき、ＨＴＭＬユニット５０は、変更されたソースツリーを参照して表示を更新する。ＳＶＧユニット６０は、自身のボキャブラリに属するノードが変更されていないので、ミューテーションイベントを無視してもよい。

【００５７】

編集の内容によっては、ＨＴＭＬユニット５０による表示の更新に伴って、全体のレイアウトが変わる可能性がある。この場合は、画面のレイアウトを管理する構成、例えば最上位のノードの表示を担当するプラグインにより、プラグインごとの表示領域のレイアウトが更新される。例えば、ＨＴＭＬユニット５０による表示領域が以前より大きくなった場合、ＨＴＭＬユニット５０は、まず自身の担当する部分を描画して、表示領域の大きさを決定する。そして、画面のレイアウトを管理する構成に、変更後の表示領域の大きさを通知し、レイアウトの更新を依頼する。画面のレイアウトを管理する構成は、通知を受けて、プラグインごとの表示領域を再レイアウトする。こうして、編集された部分の表示が適切に更新されるとともに、画面全体のレイアウトが更新される。

【００５８】

つづいて、実施の形態の文書処理装置２０を実現する機能構成について更に詳細に説明

する。

【００５９】

文書を処理し管理するシステムの実施の形態が図１１－２９を参照して説明される。

【００６０】

図１１（ａ）は、後述するタイプの文書処理／管理システムの基礎として機能する構成の従来の配置例を示す。構成１０は、通信経路１３によりメモリ１２に接続されたＣＰＵ又はマイクロプロセッサ１１などの形式のプロセッサを含む。メモリ１２は、現在又は将来に利用可能な任意のＲＯＭ及び／又はＲＡＭの形式であってもよい。通信経路１３は、典型的にはバスとして設けられる。マウス、キーボード、音声認識システムなどのユーザ入力装置１４及び表示装置１５（又は他のユーザインタフェイス）に対する入出力インタフェイス１６も、プロセッサ１１とメモリ１２の通信のためのバスに接続される。よく知られているように、プリンタ、通信モデムなどの他の装置が結合されてもよい。この構成は、スタンドアロンであってもよいし、複数の端末及び１以上のサーバが接続されてネットワーク化された形式であってもよいし、既知のいかなる方式により構成されてもよい。本発明は、これらのコンポーネントの配置、集中又は分配されたアーキテクチャー、あるいは様々なコンポーネントの通信方法により制限されない。

【００６１】

さらに、本システム及びここで議論される実施例は、様々な機能性を提供するいくつかのコンポーネント及びサブコンポーネントを含むものとして議論されることに注意されたい。これらのコンポーネント及びサブコンポーネントは、注目された機能性を提供するために、ハードウェアとソフトウェアの組合せだけでなく、ハードウェアのみ、ソフトウェアのみによっても実現されうる。さらに、ハードウェア、ソフトウェア、及びそれらの組合せは、汎用の計算装置、専用のハードウェア、又はそれらの組合せにより実現されうる。したがって、コンポーネント又はサブコンポーネントの構成は、コンポーネント又はサブコンポーネントの機能性を提供するための特定のソフトウェアを実行する汎用／専用の計算装置を含む。

【００６２】

図１１（ｂ）は、文書処理／管理システムの一例の全体のブロック図を示す。このような文書処理／管理システムにおいて文書が生成され編集される。これらの文書は、例えばＸＭＬなど、マークアップ言語の特徴を有する任意の言語により記述されてもよい。また、便宜上、特定のコンポーネント及びサブコンポーネントの用語及びタイトルを創造した。しかしながら、これらは、この開示の一般的な教示の範囲を制限するために解釈されるべきではない。

【００６３】

文書処理／管理システムは、２つの基本的な構成を有するものととらえることができる。第１の構成は、文書処理／管理システムが動作する環境である「実行環境」１０１である。例えば、実行環境は、文書の処理中及び管理中に、ユーザだけでなくシステムも支援する、基本的なユーティリティ及び機能を提供する。第２の構成は、実行環境において走るアプリケーションから構成される「アプリケーション」１０２である。これらのアプリケーションは、文書自身及び文書の様々な表現を含む。

【００６４】

１．実行環境

実行環境１０１のキーとなる構成はProgramInvoker（プログラムインボーカ：プログラム起動部）１０３である。ProgramInvoker１０３は、文書処理／管理システムを起動するためにアクセスされる基本的なプログラムである。例えば、ユーザが文書処理／管理システムにログオンして開始するとき、ProgramInvoker１０３が実行される。ProgramInvoker１０３は、例えば、文書処理／管理システムにプラグインとして加えられた機能を読み出して実行させたり、アプリケーションを開始して実行させたり、文書に関連する特性を読み出すことができる。ProgramInvoker１０３の機能はこれらに限定されない。ユーザが実行環境内で実行されるように意図されたアプリケーションを起動したいとき、ProgramInv

oker１０３は、そのアプリケーションを見つけ、それを起動して、アプリケーションを実行する。例えば、ユーザがシステム上に既にロードされたドキュメント（それは実行環境中のアプリケーションである）を編集したい場合、ProgramInvoker１０３は、最初に文書を見つけて、次に、文書をロードし編集するために必要な機能を実行する。

【００６５】

ProgramInvoker１０３には、プラグインサブシステム１０４、コマンドサブシステム１０５、及びResource（リソース）モジュール１０９などのいくつかの構成がアタッチされる。これらの構成については、以下に詳述する。

【００６６】

１．ａ．プラグインサブシステム

プラグインサブシステム１０４は、文書処理／管理システムに機能を追加するための高度に柔軟で効率的な構成として使用される。プラグインサブシステム１０４は、また、文書処理／管理システムに存在する機能を修正又は削除するために使用することができる。さらに、種々様々の機能をプラグインサブシステムを使用して追加又は修正することができる。例えば、上述され、かつ後で詳述する画面上への文書の描画を支援するように作用するEditlet（エディットレット：編集部）機能の追加を要望されてもよい。Editletプラグインは、システムに追加されるボキャブラリの編集も支援する。

【００６７】

プラグインサブシステム１０４は、ServiceBroker（サービスブローカ：サービス仲介部）１０４１を含む。ServiceBroker１０４１は、文書処理／管理システムに加えられるプラグインを管理することにより、文書処理／管理システムに加えられるサービスを仲介する。

【００６８】

所望の機能性を実現する個々の機能は、Service（サービス）１０４２の形でシステムに追加される。利用可能なService１０４２のタイプは、Application（アプリケーション）サービス、ZoneFactory（ゾーンファクトリ：ゾーン生成部）Service、Editlet（エディットレット：編集部）Service、CommandFactory（コマンドファクトリ：コマンド生成部）Service、ConnectXPath（コネクトＸＰａｔｈ：ＸＰａｔｈ管理部）Service、ＣＳＳComputation（ＣＳＳコンピューテーション：ＣＳＳ計算部）Serviceなどを含むが、これらに限定されない。これらのService、及びシステムの他の構成とそれらとの関係は、文書処理／管理システムについてのよりよい理解のために、以下に詳述される。

【００６９】

プラグインとServiceの関係は以下の通りである。プラグインは、１以上のServiceProvider（サービスプロバイダ：サービス提供部）を含むことができるユニットである。それぞれのServiceProviderは、それに関連したServiceの１以上のクラスを有する。例えば、適切なソフトウェアアプリケーションを有する単一のプラグインを使用することにより、１以上のServiceをシステムに追加することができ、これにより、対応する機能をシステムに追加することができる。Editletサービスなど、既存のサービスであっても、単一又は複数のボキャブラリを処理する能力がそれぞれのプラグインにより提供されてもよい。

【００７０】

１．ｂ．コマンドサブシステム

コマンドサブシステム１０５は、文書の処理に関連したコマンドの形式の命令を実行するために使用される。ユーザは、一連の命令を実行することにより、文書に対する操作を実行することができる。例えば、ユーザは、コマンドの形で命令を発行することにより、文書処理／管理システム中のＸＭＬ文書に対応するＸＭＬのＤＯＭツリーを編集し、ＸＭＬ文書を処理する。これらのコマンドは、キーストローク、マウスクリック、又は他の有効なユーザインタフェイスアクションを使用して入力されてもよい。１つのコマンドにより１以上の命令が実行されることもある。この場合、これらの命令が１つのコマンドにラップ（包含）され、連続して実行される。例えば、ユーザが、誤った単語を正しい単語に置換したいとする。この場合、第１の命令は、文書中の誤った単語を発見することであり

、第２の命令は、誤った単語を削除することであり、第３の命令は、正しい単語を挿入することであろう。これらの３つの命令が１つのコマンドにラップされてもよい。

【００７１】

いくつかの例においては、コマンドは、関連した機能、例えば、後で詳述する「アンドゥ」機能を有してもよい。これらの機能は、オブジェクトを生成するために使用されるいくつかの基本クラスにも割り当てられてもよい。

【００７２】

コマンドサブシステム１０５のキーとなる構成は、選択的にコマンドを示し実行するように作用するCommandInvoker（コマンドインボーカ：コマンド起動部）１０５１である。図１１（ｂ）には、１つのCommandInvokerのみが示されているが、１以上のCommandInvokerが使用されてもよく、１以上のコマンドが同時に実行されてもよい。CommandInvoker１０５１は、コマンドを実行するために必要な機能及びクラスを保持する。動作において、実行されるべきCommand（コマンド：命令）１０５２は、Queue（キュー）１０５３に積まれる。CommandInvokerは、連続的に実行するコマンドスレッドを生成する。CommandInvoker内で既に実行中のCommandがなければ、CommandInvoker１０５１により実行されるように意図されたCommand１０５２が実行される。CommandInvokerが既にコマンドを実行している場合、新しいCommandは、Queue１０５３の最後に積まれる。しかしながら、それぞれのCommandInvoker１０５１では、一度に１つのCommandのみが実行される。指定されたCommandの実行に失敗した場合、CommandInvoker１０５１は例外処理を実行する。

【００７３】

CommandInvoker１０５１により実行されるCommandの型は、UndoableCommand（取消可能コマンド）１０５４、AsyncronousCommand（非同期コマンド）１０５５、及びVCCommand（ＶＣコマンド）１０５６を含むが、これらに限定されない。UndoableCommand１０５４は、ユーザが望めば、そのCommandの結果を取り消すことが可能なCommandである。UndoableCommandの例として、切り取り、コピー、テキストの挿入、などがある。動作において、ユーザが文書の一部を選択し、その部分に切り取りコマンドを適用するとき、UndoableCommandを用いることにより、切り取られた部分は、必要であれば、「切り取られていない」ようにすることができる。

【００７４】

VCCommand１０５６は、ボキャブラリコネクション記述子（Vocabulary Connection Discriptor：ＶＣＤ）スクリプトファイルに格納される。これらは、プログラマにより定義されうるユーザ指定のCommandである。Commandは、例えば、ＸＭＬフラグメントを追加したり、ＸＭＬフラグメントを削除したり、属性を設定したりするための、より抽象的なCommandの組合せであってもよい。これらのCommandは、特に、文書の編集に焦点を合わせている。

【００７５】

AsyncronousCommand１０５５は、文書のロードや保存など、システムよりのCommandであり、UndoableCommandやVCCommandとは別に、非同期的に実行される。AsyncronousCommandは、UndoableCommandではないので、取り消すことはできない。

【００７６】

１．ｃ．リソース

Resource１０９は、様々なクラスに、いくつかの機能を提供するオブジェクトである。例えば、ストリングリソース、アイコン、及びデフォルトキーバインドは、システムで使用されるResourceの例である。

【００７７】

２．アプリケーションコンポーネント

文書処理／管理システムの第２の主要な特徴であるアプリケーションコンポーネント１０２は、実行環境１０１において実行される。アプリケーションコンポーネント１０２は、実際の文書と、システム内における文書の様々な論理的、物理的な表現を含む。さらに、アプリケーションコンポーネント１０２は、文書を管理するために使用されるシステム

の構成を含む。アプリケーションコンポーネント１０２は、さらに、UserApplication（ユーザアプリケーション）１０６、アプリケーションコア１０８、ユーザインタフェイス１０７、及びCoreComponent（コアコンポーネント）１１０を含む。

【００７８】

２．ａ．ユーザアプリケーション

UserApplication１０６は、ProgramInvoker１０３と共にシステム上にロードされる。UserApplication１０６は、文書と、文書の様々な表現と、文書と対話するために必要なユーザインタフェイスとをつなぐ接着剤となる。例えば、ユーザが、プロジェクトの一部である文書のセットを生成したいとする。これらの文書がロードされると、文書の適切な表現が生成される。ユーザインタフェイス機能は、UserApplication１０６の一部として追加される。言いかえれば、UserApplication１０６は、ユーザがプロジェクトの一部を形成する文書と対話することを可能とする文書の表現と、文書の様々な態様とを、共に保持する。一旦UserApplication１０６が生成されると、ユーザがプロジェクトの一部を形成する文書との対話を望むたびに、ユーザは簡単に実行環境上にUserApplication１０６をロードすることができる。

【００７９】

２．ｂ．コアコンポーネント

CoreComponent１１０は、複数のPane（ペイン）の間で文書を共有する方法を提供する。後で詳述するように、Paneは、ＤＯＭツリーを表示し、画面の物理的なレイアウトを扱う。例えば、物理的な画面は、個々の情報の断片を描写する画面内の複数のPaneからなる。ユーザから画面上に見える文書は、１又はそれ以上のPaneに出現しうる。また、２つの異なる文書が画面上で２つの異なるPaneに現れてもよい。

【００８０】

図１１（ｃ）に示されるように、画面の物理的なレイアウトもツリーの形式になっている。コンポーネント１０８３がPaneとして画面上に存在するとき、Paneは、RootPane（ルートペイン）１０８４としても実現され得るし、SubPane（サブペイン）１０８５でもあり得る。RootPane１０８４は、Paneのツリーの根に当たるPaneであり、SubPane１０８５は、RootPane１０８４以外の任意のPaneである。

【００８１】

CoreComponent１１０は、さらに、フォントを提供し、ツールキットなど、文書のための複数の機能的な操作のソースの役割を果たす。CoreComponent１１０により実行されるタスクの一例に、複数のPane間におけるマウスカーソルの移動がある。実行されるタスクの他の例として、あるPane中の文書の一部をマークし、それを異なる文書を含む別のPane上にコピーする。

【００８２】

２．ｃ．アプリケーションコア

上述したように、アプリケーションコンポーネント１０２は、システムにより処理され管理される文書から構成される。これは、システム内における文書の様々な論理的及び物理的な表現を含む。アプリケーションコア１０８は、アプリケーションコンポーネント１０２の構成である。その機能性は、実際の文書を、それに含まれる全てのデータとともに保持することである。アプリケーションコア１０８は、DocumentManager（ドキュメントマネージャ：文書管理部）１０８１及びDocument（ドキュメント：文書）１０８２自身を含む。

【００８３】

DocumentManager１０８１の様々な態様を以下に詳述する。DocumentManager１０８１は、Document１０８２を管理する。DocumentManager１０８１は、RootPane１０８４、SubPane１０８５、ClipBoard（クリップボード）ユーティリティ１０８６、及びSnapShot（スナップショット）ユーティリティ１０８７にも接続される。ClipBoardユーティリティ１０８６は、ユーザがクリップボードに加えることを決定した文書の部分を保持する方法を提供する。例えば、ユーザが、文書の一部を切り取り、後で再考するために新規文書にそ

れを保存することを望んだとする。このような場合、切り取られた部分がClipBoard１０８６に追加される。

【００８４】

SnapShotユーティリティ１０８７についても次に説明する。SnapShotユーティリティ１０８７は、アプリケーションがある状態から別の状態まで移行するときに、アプリケーションの現在の状態を記憶することを可能とする。

【００８５】

２．ｄ．ユーザインタフェイス

アプリケーションコンポーネント１０２の別の構成は、ユーザがシステムと物理的に対話する手段を提供するユーザインタフェイス１０７である。例えば、ユーザインタフェイスは、ユーザが文書をアップロードしたり、削除したり、編集したり、管理したりするために使用される。ユーザインタフェイスは、Frame（フレーム）１０７１、MenuBar（メニューバー）１０７２、StatusBar（ステータスバー）１０７３、及びURLBar（ＵＲＬバー）１０７４を含む。

【００８６】

Frameは、一般に知られているように、物理的な画面のアクティブな領域であるとみなされる。MenuBar１０７２は、ユーザに対する選択を示すメニューを含む画面領域である。StatusBar１０７３は、アプリケーションの実行状態を表示する画面領域である。URLBar１０７４は、インターネットをナビゲートするためにＵＲＬアドレスを入力する領域を提供する。

【００８７】

３．文書管理及び関連するデータ構造

図１２は、DocumentManager１０８１の詳細を示す。これは、文書処理／管理システム内で文書を表現するために用いられるデータ構造及び構成を含む。分かりやすくするために、このサブセクションで説明される構成は、ＭＶＣパラダイムを用いて説明される。

【００８８】

DocumentManager１０８１は、文書処理／管理システム内にある全ての文書を保持しホストするDocumentContainer（ドキュメントコンテナ：文書コンテナ）２０３を含む。DocumentManager１０８１にアタッチされたツールキット２０１は、DocumentManager１０８１により使用される様々なツールを提供する。例えば、DomService（ＤＯＭサービス）は、文書に対応するＤＯＭを生成し、保持し、管理するために必要とされる全ての機能を提供するために、ツールキット２０１により提供されるツールである。ツールキット２０１により提供される別のツールであるIOManager（入出力管理部）は、システムへの入力及びシステムからの出力を管理する。同様に、StreamHandler（ストリームハンドラ）は、ビットストリームによる文書のアップロードを扱うツールである。これらのツールは、図中に特に示さず、参照番号を割り当てないが、ツールキット２０１のコンポーネントを形成する。

【００８９】

ＭＶＣパラダイムの表現によれば、モデル（Ｍ）は、文書のＤＯＭツリーモデル２０２を含む。前述したように、全ての文書は、文書処理／管理システムにおいてＤＯＭツリーとして表現される。文書は、また、DocumentContainer２０３の一部を形成する。

【００９０】

３．ａ．ＤＯＭモデル及びゾーン

文書を表現するＤＯＭツリーは、Node（ノード）２０２１を有するツリーである。ＤＯＭツリーの部分集合であるZone（ゾーン）２０９は、ＤＯＭツリー内の１以上のNodeの関連領域を含む。例えば、画面上で文書の一部のみを表示し得るが、この可視化された文書の一部はZone２０９を用いて表示される。Zoneは、ZoneFactory（ゾーンファクトリ：ゾーン生成部）２０５と呼ばれるプラグインを用いて、生成され、取り扱われ、処理される。ZoneはＤＯＭの一部を表現するが、１以上の「名前空間」を使用してもよい。よく知られているように、名前空間は、名前空間内でユニークな名前の集合である。換言すれば、

名前空間内に同じ名前は存在しない。

【００９１】

３．ｂ．Facet及びFacetとZoneとの関係

Facet（ファセット）２０２２は、ＭＶＣパラダイムのモデル（Ｍ）部分内の別の構成である。Facetは、ZoneにおいてNodeを編集するために使用される。Facet２０２２は、Zone自身の内容に影響を与えずに実行することができる手続（プロシージャ）を使用して、ＤＯＭへのアクセスを編成する。次に説明するように、これらの手続は、Nodeに関連した重要で有用な操作を実行する。

【００９２】

各Node２０２１は、対応するFacet２０２２を有する。ＤＯＭの中のNodeを直接操作する代わりに、操作を実行するためにFacetを使用することによって、ＤＯＭの保全性は保護される。そうでなければ、操作がNode上で直接実行される場合、いくつかのプラグインがＤＯＭを同時に変更することができ、その結果矛盾を引き起こす。

【００９３】

Ｗ３Ｃが策定したＤＯＭの標準規格は、Nodeを操作するための標準的なインタフェイスを定義するが、実際には、ボキャブラリごと又はNodeごとに特有の操作があるので、これらの操作をＡＰＩとして用意しておくのが好都合である。文書処理／管理システムでは、このような各Nodeに特有のＡＰＩをFacetとして用意し、各Nodeにアタッチする。これにより、ＤＯＭの標準規格に準拠しつつ、有用なＡＰＩを付加することができる。また、ボキャブラリごとに特有のＤＯＭを実装するのではなく、標準的なＤＯＭの実装に、後から特有のＡＰＩを付加するようにすることで、多様なボキャブラリを統一的に処理することができるとともに、複数のボキャブラリが任意の組合せで混在した文書を適切に処理することができる。

【００９４】

既に定義したように、ボキャブラリは、名前空間に属するタグ（例えばＸＭＬのタグ）のセットである。上述したように、名前空間は、ユニークな名前（ここではタグ）のセットを有する。ボキャブラリは、ＸＭＬ文書を表現するＤＯＭツリーのサブツリーとして現れる。このサブツリーはZoneを含む。特定の例においては、タグセットの境界はZoneによって定義される。Zone２０９は、ZoneFactory２０５と呼ばれるServiceを利用して生成される。上述したように、Zone２０９は、文書を表現するＤＯＭツリーの一部の内部表現である。このような文書の一部へのアクセスを提供するために、論理的な表現が要求される。この論理的表現は、文書が画面上で論理的にどのように表現されるかについてコンピュータに通知する。既に定義したように、「キャンバス」、例えばCanvas２１０は、Zoneに対応する論理的なレイアウトを提供するように作用するServiceである。

【００９５】

他方、「ペイン」、例えばPane２１１は、Canvas２１０により提供される論理的なレイアウトに対応する物理的な画面レイアウトである。実際、ユーザは表示画面上で文字や画像によって文書のレンダリングのみを見る。したがって、文書は、画面上に文字や画像を描画するプロセスにより、画面上に描写されなければならない。Pane２１１により提供される物理的なレイアウトに基づいて、文書は、Canvas２１０により画面上に描写される。

【００９６】

Zone２０９に対応するCanvas２１０は、Editlet２０６を使用して生成される。文書のＤＯＭは、Editlet２０６及びCanvas２１０を使用して編集される。元の文書の完全性を維持するために、Editlet２０６及びCanvas２１０は、Zone２０９における１以上のNodeに対応するFacet２０２２を使用する。これらのServiceは、Zone及びＤＯＭ内のNodeを直接操作しない。Facetは、ＭＶＣパラダイムの「Ｃ」コンポーネント、すなわちコントローラから、Command２０７を利用して操作される。

【００９７】

ユーザは、一般に、画面上のカーソルを移動させたり、コマンドをタイプしたりすることによって、画面と対話する。画面上の論理的なレイアウトを提供するCanvas２０１０は

、このカーソル操作を受け付ける。Canvas２０１０は、対応するアクションをFacetに実行させることができる。この関係により、カーソルサブシステム２０４は、DocumentManager１０８１に対して、ＭＶＣパラダイムのコントローラ（Ｃ）として機能する。

【００９８】

Canvas２１０は、イベントを扱うタスクも有する。例えば、Canvas２１０は、マウスクリック、フォーカス移動、及びユーザにより起こされた同様のアクションなどのイベントを扱う。

【００９９】

３．ｃ．Zone、Facet、Canvas及びPaneの間の関係の概要

文書処理／管理システム内の文書は、少なくとも４つの観点から見ることができる。すなわち、１）文書処理／管理システムにおいて文書の内容及び構造を保持するために用いられるデータ構造、２）文書の保全性に影響を与えずに文書の内容を編集する手段、３）文書の画面上の論理的なレイアウト、４）文書の画面上の物理的なレイアウト、である。Zone、Facet、Canvas及びPaneは、前述の４つの観点に相当する、文書処理／管理システムのコンポーネントをそれぞれ表す。

【０１００】

３．ｄ．アンドゥサブシステム

上述したように、文書に対するいかなる変更（例えば編集）も取消可能であることが望ましい。例えば、ユーザが編集操作を実行し、次に、その変更の取消を決定したとする。図１２に関連して、アンドゥサブシステム２１２は、文書管理部の取消可能なコンポーネントを実現する。UndoManager（アンドゥマネージャ：アンドゥ管理部）２１２１は、ユーザによって取り消される可能性のある全ての文書に対する操作を保持する。

【０１０１】

例えば、ユーザが、文書中の単語を別の単語に置換するコマンドを実行したとする。その後、ユーザは考え直し、元の単語に戻すことを決定したとする。アンドゥサブシステム２１２は、UndoableEdit２１２２を用いて、このような操作を支援する。UndoManager２１２１は、このようなUndoableEdit（アンドゥアブルエディット：取消可能な編集）２１２２の操作を保持する。操作は、単一のＸＭＬ操作形式だけでなく、ＸＨＴＭＬ、ＳＶＧ、及びＭａｔｈＭＬなどの様々な言語の文書の連続的な変更を含んでもよく、それぞれの言語における変更を取消可能としてもよい。先入れ後出し操作により、使用されたボキャブラリに関係なく、最後の変更が最初に取り消され、最後から二番目の変更が次に取り消される。このように、２以上の編集部が編集される場合であっても、統合されたアンドゥが正しい順序で実行され、自然で論理的な操作感を与えることができる。

【０１０２】

３．ｅ．カーソルサブシステム

前述したように、ＭＶＣのコントローラ部分は、カーソルサブシステム２０４を備えてもよい。カーソルサブシステム２０４は、ユーザから入力を受け付ける。これらの入力は、一般にコマンド及び／又は編集操作の性格を有している。したがって、カーソルサブシステム２０４は、DocumentManager１０８１に関連したＭＶＣパラダイムのコントローラ（Ｃ）部分であると考えることができる。

【０１０３】

３．ｆ．ビュー

前述したように、Canvas２１０は、画面上に提示されるべき文書の論理的なレイアウトを表す。ＸＨＴＭＬ文書の例では、Canvas２１０は、文書が画面上でいかに見えるかを論理的に表現したボックスツリー２０８を含んでもよい。このボックスツリー２０８は、文書管理部１０８１に関連したＭＶＣパラダイムのビュー（Ｖ）部分に含まれよう。

【０１０４】

４．ボキャブラリコネクション

文書処理／管理システムの重要な特徴は、ＸＭＬ文書が、２つの異なる方法（例えば、２つのマークアップ言語）で表現され表示可能であり、２つの異なる表現の間で自動的に

整合性が維持されることである。

【０１０５】

マークアップ言語により記述された文書、例えばＸＭＬ文書は、文書型定義により定義されたボキャブラリに基づいて作成されている。ボキャブラリは、タグのセットである。ボキャブラリは、任意に定義されてもよいため、無限に多くのボキャブラリが存在しうる。しかしながら、多数の可能なボキャブラリのそれぞれに対して専用の処理／管理環境を提供するのは現実的ではない。ボキャブラリコネクションは、この問題を解決する方法を提供する。

【０１０６】

例えば、文書は２以上のマークアップ言語により記述されてもよい。文書は、例えば、ＸＨＴＭＬ（eXtensible HyperText Markup Language）、ＳＶＧ（Scalable Vector Graphics）、ＭａｔｈＭＬ（Mathematical Markup Language）、その他のマークアップ言語により記述されてもよい。換言すれば、マークアップ言語は、ＸＭＬにおけるボキャブラリやタグセットと同様に見なされてもよい。

【０１０７】

ボキャブラリは、ボキャブラリプラグインを用いて処理される。文書処理／管理システムにおいてプラグインが利用不可能であるボキャブラリにより記述された文書は、プラグインが利用可能である別のボキャブラリの文書にマッピングすることにより表示される。この特徴により、プラグインが用意されていないボキャブラリの文書も適切に表示することができる。

【０１０８】

ボキャブラリコネクションは、定義ファイルを取得し、定義ファイルの間でマッピングし、定義ファイルを生成する能力を含む。あるボキャブラリで記述された文書は、別のボキャブラリにマッピングすることができる。このように、ボキャブラリコネクションは、文書がマッピングされるボキャブラリに対応した表示／編集プラグインにより文書を表示し編集することを可能にする。

【０１０９】

上述したように、各文書は、一般に複数のノードを有するＤＯＭツリーとして文書処理／管理システムにおいて記述される。「定義ファイル」は、それぞれのノードについて、そのノードと他のノードとの対応を記述する。各ノードの要素値及び属性値が編集可能か否かが指定される。ノードの要素値又は属性値を用いた演算式が記述されてもよい。

【０１１０】

マッピングという特徴を利用して、定義ファイルを適用したデスティネーションＤＯＭツリーが生成される。このように、ソースＤＯＭツリーとデスティネーションＤＯＭツリーの関係が構築され保持される。ボキャブラリコネクションは、ソースＤＯＭツリーとデスティネーションＤＯＭツリーの対応を監視する。ユーザから編集指示を受けると、ボキャブラリコネクションは、ソースＤＯＭツリーの関連したノードを変更する。ソースＤＯＭツリーが変更されたことを示す「ミューテーションイベント」が発行され、デスティネーションＤＯＭツリーがそれに応じて変更される。

【０１１１】

ボキャブラリコネクションの使用により、少数のユーザのみに知られていた比較的マイナーなボキャブラリを、別のメジャーなボキャブラリに変換することができる。したがって、少数のユーザによって利用されるマイナーなボキャブラリであっても、文書を適切に表示し、望ましい編集環境を提供することができる。

【０１１２】

このように、文書処理／管理システムの一部であるボキャブラリコネクションサブシステムは、文書の複数の表現を可能にする機能を提供する。

【０１１３】

図１３は、ボキャブラリコネクション（ＶＣ：Vocabulary Connection）サブシステム３００を示す。ＶＣサブシステム３００は、同一の文書の２つの代替表現の整合性を維持

する方法を提供する。図中、既に図示され識別されたものと同じ構成が設けられ、その目的を達成するために相互に接続される。例えば、２つの表現は、同一文書の、２つの異なるボキャブラリによる表現であってもよい。前述したように、一方はソースＤＯＭツリーであってもよく、他方はデスティネーションＤＯＭツリーであってもよい。

【０１１４】

４．ａ．ボキャブラリコネクションサブシステム

ボキャブラリコネクションサブシステム３００の機能は、VocabularyConnection３０１と呼ばれるプラグインを使用して、文書処理／管理システムにおいて実現される。文書が表現されるVocabulary３０５ごとに、対応するプラグインが要求される。例えば、文書の一部がＨＴＭＬで記述され、残りがＳＶＧで記述されている場合、ＨＴＭＬとＳＶＧに対応するボキャブラリプラグインが要求される。

【０１１５】

VocabularyConnectionプラグイン３０１は、適切なVocabulary３０５の文書に対応した、Zone２０９又はPane２１１のための適切なVCCanvas（ボキャブラリコネクションキャンバス）３１０を生成する。VocabularyConnection３０１を用いて、ソースＤＯＭツリー内のZone２０９に対する変更は、変換ルールにより、別のＤＯＭツリー３０６の対応するZoneに伝達される。変換ルールは、ボキャブラリコネクション記述子（Vocabulary Connection Descriptor：ＶＣＤ）の形式で記述される。このようなソースＤＯＭとデスティネーションＤＯＭの間の変換に対応するそれぞれのＶＣＤファイルについて、対応するVCManager（ボキャブラリコネクションマネージャ）３０２が生成される。

【０１１６】

４．ｂ．Connector

Connector３０４は、ソースＤＯＭツリーのソースノードと、デスティネーションＤＯＭツリーのデスティネーションノードとを接続する。Connector３０４は、ソースＤＯＭツリー中のソースノード、及びソースノードに対応するソース文書に対する修正（変更）を見るために作用する。そして、対応するデスティネーションＤＯＭツリーのノードを修正する。Connector３０４は、デスティネーションＤＯＭツリーを修正することができる唯一のオブジェクトである。例えば、ユーザは、ソース文書、及び対応するソースＤＯＭツリーに対してのみ修正を行うことができる。その後、Connector３０４がデスティネーションＤＯＭツリーに、対応する修正を行う。

【０１１７】

Connector３０４は、図１３に示されるように、ツリー構造を形成するために、論理的にリンクされる。Connector３０４により形成されたツリーは、ConnectorTree（コネクタツリー）と呼ばれる。Connector３０４は、ConnectorFactory（コネクタファクトリ：コネクタ生成部）３０３と呼ばれるServiceを用いて生成される。ConnectorFactory３０３は、ソース文書からConnector３０４を生成し、それらをリンクしてConnectorTreeを形成する。VocabularyConnectionManager３０２は、ConnectorFactory３０３を保持する。

【０１１８】

前述したように、ボキャブラリは名前空間中のタグのセットである。図１３に示されるように、Vocabulary３０５は、VocabularyConnection３０１によって文書に対して生成される。これは、文書ファイルを解析し、ソースＤＯＭとデスティネーションＤＯＭの間の写像のための適切なVocabularyConnectionManager３０２を生成することにより行われる。さらに、Connectorを生成するConnectorFactory３０３と、Zone２０９を生成するZoneFactory２０５と、Zone内のノードに対応するCanvasを生成するEditlet２０６との間の適切な関係が作られる。ユーザがシステムから文書を処分又は削除するとき、対応するVocabularyConnectionManager３０２が削除される。

【０１１９】

Vocabulary３０５は、VCCanvas３１０を生成する。さらに、Connector３０４及びデスティネーションＤＯＭツリー３０６が対応して生成される。

【０１２０】

ソースＤＯＭ及びCanvasは、それぞれ、モデル（Ｍ）及びビュー（Ｖ）に対応する。しかしながら、このような表現は、ターゲットのボキャブラリが画面上に描写可能である場合に限って意味がある。描写は、ボキャブラリプラグインにより行われる。ボキャブラリプラグインは、主要なボキャブラリ、例えば、ＸＨＴＭＬ、ＳＶＧ、ＭａｔｈＭＬについて提供される。ターゲットのボキャブラリに関してボキャブラリプラグインが使用される。これらは、ボキャブラリコネクション記述子を用いてボキャブラリ間でマッピングする方法を提供する。

【０１２１】

このようなマッピングは、ターゲットのボキャブラリが、マッピング可能で、画面上に描写される方法が予め定義されたものである場合にのみ意味がある。このようなレンダリング方法は、例えばＸＨＴＭＬなどのように、Ｗ３Ｃなどの組織により定義された標準規格となっている。

【０１２２】

ボキャブラリコネクションが必要であるとき、VCCanvasが使用される。この場合、ソースのビューを直接生成することができないので、ソースのCanvasは生成されない。この場合、VCCanvasが、ConnectorTreeを使用して生成される。このVCCanvasは、イベントの変換のみを扱い、画面上の文書の描写を援助しない。

【０１２３】

４．ｃ．DestinationZone、Pane、及びCanvas

上述したように、ボキャブラリコネクションサブシステムの目的は、同一の文書の２つの表現を同時に生成し保持することである。第２の表現も、ＤＯＭツリーの形式であり、これはデスティネーションＤＯＭツリーとして既に説明した。第２の表現における文書を見るために、DestinationZone、Canvas及びPaneが必要である。

【０１２４】

VCCanvasが作成されると、図１３に示すように、対応するDestinationPane３０７が生成される。さらに、関連するDestinationCanvas３０８と、対応するBoxTree３０９が生成される。同様に、VCCanvas３１０も、ソース文書に対するPane２１１及びZone２０９に関連づけられる。

【０１２５】

DestinationCanvas３０８は、第２の表現における文書の論理的なレイアウトを提供する。特に、DestinationCanvas３０８は、デスティネーション表現における文書を描写するために、カーソルや選択のようなユーザインタフェイス機能を提供する。DestinationCanvas３０８に生じたイベントは、Connectorに供給される。DestinationCanvas３０８は、マウスイベント、キーボードイベント、ドラッグアンドドロップイベント、及び文書のデスティネーション（第２）表現のボキャブラリに特有なイベントを、Connector３０４に通知する。

【０１２６】

４．ｄ．ボキャブラリコネクションコマンドサブシステム

図１３に示したボキャブラリコネクション（ＶＣ）サブシステム３００の要素として、ボキャブラリコネクション（ＶＣ）コマンドサブシステム３１３がある。ボキャブラリコネクションコマンドサブシステム３１３は、ボキャブラリコネクションサブシステム３００に関連した命令の実行のために使用されるVCCommand（ボキャブラリコネクションコマンド）３１５を生成する。VCCommandは、内蔵のCommandTemplate（コマンドテンプレート）３１３１を使用して、及び／又は、スクリプトサブシステム３１４においてスクリプト言語を使用してスクラッチからコマンドを生成することにより、生成することができる。

【０１２７】

コマンドテンプレートには、例えば、「If」コマンドテンプレート、「When」コマンドテンプレート、「挿入（Insert）」コマンドテンプレートなどがある。これらのテンプレートは、VCCommandを作成するために使用される。

【０１２８】

４．ｅ．Ｘｐａｔｈサブシステム

Ｘｐａｔｈサブシステム３１６は、文書処理／管理システムの重要な構成であり、ボキャブラリコネクションの実現を支援する。Connector３０４は、一般にxpath情報を含む。上述したように、ボキャブラリコネクションのタスクの１つは、ソースＤＯＭツリーの変化をデスティネーションＤＯＭツリーに反映させることである。xpath情報は、変更／修正を監視されるべきソースＤＯＭツリーのサブセットを決定するために用いられる１以上のxpath表現を含む。

【０１２９】

４．ｆ．ソースＤＯＭツリー、デスティネーションＤＯＭツリー、及びConnectorTreeの概要

ソースＤＯＭツリーは、別のボキャブラリに変換される前のボキャブラリで文書を表現したＤＯＭツリー又はZoneである。ソースＤＯＭツリーのノードは、ソースノードと呼ばれる。

【０１３０】

それに対して、デスティネーションＤＯＭツリーは、ボキャブラリコネクションに関連して前述したように、同一の文書を、マッピングにより変換された後の異なるボキャブラリで表現したＤＯＭツリー又はZoneである。デスティネーションＤＯＭツリーのノードは、デスティネーションノードと呼ばれる。

【０１３１】

ConnectorTreeは、ソースノードとデスティネーションノードの対応を表すConnectorに基づく階層的表現である。Connectorは、ソースノードと、ソース文書になされた修正を監視し、デスティネーションＤＯＭツリーを修正する。Connectorは、デスティネーションＤＯＭツリーを修正することを許された唯一のオブジェクトである。

【０１３２】

５．文書処理／管理システムにおけるイベントフロー

実用のためには、プログラムはユーザからのコマンドに応答しなければならない。イベントは、プログラム上で実行されたユーザアクションを記述し実行する方法である。多くの高級言語、例えばＪａｖａ（登録商標）は、ユーザアクションを記述するイベントに頼っている。従来、プログラムは、ユーザアクションを理解し、それを自身で実行するために、積極的に情報を集める必要があった。これは、例えば、プログラムが自身を初期化した後、ユーザが画面、キーボード、マウスなどでアクションを起こしたときに適切な処理を講じるために、ユーザのアクションを繰り返し確認するループに入ることを意味する。しかしながら、このプロセスは扱いにくい。さらに、それは、ユーザが何かをするのを待つ間、ＣＰＵサイクルを消費してループするプログラムを必要とする。

【０１３３】

多くの言語が、異なるパラダイムを採用することにより、これらの問題を解決している。そのうちの一つは、現代の全てのウィンドウシステムの基礎となっている、イベントドリブンプログラミングである。このパラダイムでは、全てのユーザアクションは、「イベント」と呼ばれる抽象的な事象の集合に属する。イベントは、十分詳細に、特定のユーザアクションを記述する。プログラムがユーザにより生成されたイベントを積極的に収集するのではなく、監視すべきイベントが生じたときに、システムがプログラムに通知する。この方法によりユーザとの対話を扱うプログラムは「イベントドリブン」であると言われる。

【０１３４】

これは、多くの場合、ユーザにより生成された全てのイベントの基本特性を獲得する「Event（イベント）」クラスを使用して扱われる。

【０１３５】

文書処理／管理システムは、自身のイベント、及びこれらのイベントを扱う方法を定義して使用する。いくつかの型のイベントが使用される。例えば、マウスイベントは、ユーザのマウスアクションから起こるイベントである。マウスを含むユーザアクションは、Ca

nvas２１０によって、マウスイベントに渡される。このように、Canvasは、システムのユーザによる相互作用の最前部にあると言える。必要であれば、最前部にあるCanvasは、そのイベントに関連した内容を子へ渡す。

【０１３６】

それに対して、キーストロークイベントは、Canvas２１０から流れる。キーストロークイベントは、即時的な焦点を有する。すなわち、それは、いかなる瞬間でも作業に関連する。Canvas２１０上に入力されたキーストロークイベントは、その親に渡される。キー入力は、文字列挿入を扱うことが可能な、異なるイベントによって処理される。文字列の挿入を扱うイベントは、キーボードを使用して文字が挿入されたときに発生する。他の「イベント」は、例えば、ドラッグイベント、ドロップイベント、マウスイベントと同様に扱われる他のイベントを含む。

【０１３７】

５．ａ．ボキャブラリコネクション外のイベントの取り扱い

イベントは、イベントスレッドを用いて渡される。Canvas２１０は、イベントを受け取ると、その状態を変更する。必要であれば、Command１０５２がCanvas２１０によりCommandQueue１０５３にポストされる。

【０１３８】

５．ｂ．ボキャブラリコネクション内のイベントの取り扱い

VocabularyConnectionプラグイン３０１を用いて、DestinationCanvas１１０６は、発生したイベント、例えば、マウスイベント、キーボードイベント、ドラッグアンドドロップイベント、及びボキャブラリに特有のイベントなどを受け取る。これらのイベントは、コネクタ１１０４に通知される。より詳細には、図２１に図示されるように、VocabularyConnectionプラグイン３０１内のイベントフローは、SourcePane１１０３、VCCanvas１１０４、DestinationPane１１０５、DestinationCanvas１１０６、デスティネーションＤＯＭツリー及びConnectorTreeを通過する。

【０１３９】

６．ProgramInvoker及びProgramInvokerと他の構成との関係

ProgramInvoker１０３及びそれと他の構成との関係は、図１４（ａ）に更に詳細に示される。ProgramInvoker１０３は、文書処理／管理システムを開始するために実行される実行環境中の基本的なプログラムである。図１４（ｂ）に図示されるように、UserApplication１０６、ServiceBroker１０４１、CommandInvoker１０５１、及びResource１０９は、全てProgramInvoker１０３に接続される。前述したように、アプリケーション１０２は、実行環境中で実行されるコンポーネントである。同様に、ServiceBroker１０４１は、システムに様々な機能を加えるプラグインを管理する。他方、CommandInvoker１０５１は、ユーザにより提供される命令を実行して、コマンドを実行するために使用されるクラス及びファンクションを保持する。

【０１４０】

６．ａ．プラグイン及びサービス

ServiceBroker１０４１について、図１４（ｂ）を参照して更に詳細に説明する。前述したように、CommandInvoker１０４１は、システムに様々な機能を追加するプラグイン（及び関連するサービス）を管理する。Service１０４２は、文書処理／管理システムに特徴を追加又は変更可能な最も下の層である。「Service」は、ServiceCategory４０１とServiceProvider４０２の２つの部分からなる。図１４（ｃ）に図示されるように、１つのServiceCategory４０１は、複数の関連するServiceProvider４０２を持ちうる。それぞれのServiceProviderは、特定のServiceCategoryの一部または全部を実行するように作用する。ServiceCategory４０１は、他方では、Serviceの型を定義する。

【０１４１】

Serviceは、１）文書処理／管理システムに特定の特色を提供する「特色サービス」、２）文書処理／管理システムにより実行されるアプリケーションである「アプリケーションサービス」、３）文書処理／管理システムの全体にわたって必要な特色を提供する「環

境サービス」、の３つの型に分類することができる。

【０１４２】

Serviceの例は、図１４（ｄ）に示される。アプリケーションServiceのCategoryにおいては、システムユーティリティが対応するServiceProviderの例である。同様に、Editlet２０６はCategoryであり、HTMLEditlet及びSVGEditletは対応するServiceProviderである。ZoneFactory２０５は、Serviceの別のCategoryであり、対応するServiceProvider（図示せず）を有する。

【０１４３】

プラグインは、文書処理／管理システムに機能性を加えると既に説明したが、図１４（ｃ）及び（ｄ）に示すように、いくつかのServiceProvider４０２及びそれらに関連するクラスからなるユニットと見なされてもよい。各プラグインは、宣言ファイルに記述された依存性及びServiceCategory４０１を有するだろう。

【０１４４】

６．ｂ．プログラムインボーカとアプリケーションとの関係

図１４（ｅ）は、ProgramInvoker１０３とUserApplication１０６との関係についての更なる詳細を示す。必要な文書やデータなどは、ストレージからロードされる。必要なプラグインは、全てServiceBroker１０４１上にロードされる。ServiceBroker１０４１は、全てのプラグインを保持し管理する。プラグインは、システムに物理的に追加することができ、又、その機能はストレージからロードすることができる。プラグインの内容がロードされると、ServiceBroker１０４１は、対応するプラグインを定義する。つづいて、対応するUserApplication１０６が生成され、実行環境１０１にロードされ、ProgramInvoker１０３にアタッチされる。

【０１４５】

７．アプリケーションサービスと環境との関係

図１５（ａ）は、ProgramInvoker１０３上にロードしたアプリケーションサービスの構成についての更なる詳細を示す。コマンドサブシステム１０５のコンポーネントであるCommandInvoker１０５１は、ProgramInvoker１０３内のCommand１０５２を起動又は実行する。Command１０５２は、文書処理／管理システムにおいて、ＸＭＬなどの文書を処理し、対応するＸＭＬＤＯＭツリーを編集するために用いられる命令である。CommandInvoker１０５１は、Command１０５２を実行するために必要なクラス及びファンクションを保持する。

【０１４６】

ServiceBroker１０４１も、ProgramInvoker１０３内で実行される。UserApplication１０６は、ユーザインタフェイス１０７及びCoreComponent１１０に接続される。CoreComponent１１０は、全てのPaneの間で文書を共有する方法を提供する。CoreComponent１１０は、さらにフォントを提供し、Paneのためのツールキットの役割を果たす。

【０１４７】

図１５（ａ）（ｂ）は、Frame１０７１、MenuBar１０７２、及びStatusBar１０７３の関係を示す。

【０１４８】

８．アプリケーションコア

図１６（ａ）は、全ての文書、及び文書の一部及び文書に属するデータを保持するアプリケーションコア１０８についての更なる説明を提供する。CoreComponent１１０は、文書１０８２を管理するDocumentManager１０８１にアタッチされる。DocumentManager１０８１は、文書処理／管理システムに関連づけられたメモリに格納される全ての文書１０８２の所有者である。

【０１４９】

画面上の文書の表示を容易にするために、DocumentManager１０８１はRootPane１０８４にも接続される。ClipBoard１０８５、SnapShot１０８７、Drag&Drop６０１、及びOverlay６０２の機能も、CoreComponent１１０にアタッチされる。

【０１５０】
SnapShot１０８７は、図１６（ｂ）に示すように、アプリケーションの状態を元に戻すために使用される。ユーザがSnapShot１０８７を起動したとき、アプリケーションの現状が検知され、格納される。その後、アプリケーションの状態が別の状態に変わるとき、格納された状態の内容は保存される。SnapShot１０８７は、図１６（ｂ）に図示される。動作において、アプリケーションがあるＵＲＬから他へ移動するときに、前に戻る動作及び先に進む動作をシームレスに実行可能とするために、SnapShot１０８７は以前の状態を記憶する。
【０１５１】
９．DocumentManager内における文書の構成
図１７（ａ）は、DocumentManager１０８１の更なる説明と、DocumentManagerにおいて文書が構成され保持される様子を示す。図１７（ｂ）に示したように、DocumentManager１０８１は、文書１０８２を管理する。図１７（ａ）に示される例において、複数の文書のうちの１つはRootDocument（ルート文書）７０１であり、残りの文書はSubDocument（サブ文書）７０２である。DocumentManager１０８１は、RootDocument７０１に接続され、RootDocument７０１は、全てのSubDocument７０２に接続される。
【０１５２】
図１２及び図１７（ａ）に示すように、DocumentManager１０８１は、全ての文書１０８２を管理するオブジェクトであるDocumentContainer２０３に結合される。DOMService７０３及びIOManager７０４を含むツールキット２０１（例えばＸＭＬツールキット）の一部を形成するツールも、DocumentManager１０８１に供給される。再び図１７（ａ）を参照して、DOMService７０３は、DocumentManager１０８１により管理される文書に基づいたＤＯＭツリーを生成する。各Document７０５は、それがRootDocument７０１であってもSubDocument７０２であっても、対応するDocumentContainer２０３によって管理される。
【０１５３】
図１７（ｂ）は、文書Ａ－Ｅが階層的に配置される様子を示す。文書ＡはRootDocumentである。文書Ｂ－Ｄは、文書ＡのSubDocumentである。文書Ｅは、文書ＤのSubDocumentである。図１７（ｂ）は、これと同じ文書の階層が画面上に表示された例も示す。RootDocumentである文書Ａは、基本フレームとして表示される。文書ＡのSubDocumentである文書Ｂ－Ｄは、基本フレームＡの中のサブフレームとして表示される。文書ＤのSubDocumentである文書Ｅは、サブフレームＤのサブフレームとして画面に表示される。
【０１５４】
再び図１７（ａ）を参照して、UndoManager（アンドゥマネージャ：アンドゥ管理部）７０６及びUndoWrapper（アンドゥラッパー）７０７は、それぞれのDocumentContainer２０３に対して生成される。UndoManager７０６及びUndoWrapper７０７は、取消可能なコマンドを実行するために使用される。この特徴を使用することにより、編集操作を使用して文書に対して実行された変更を取り消すことができる。SubDocumentの変更は、RootDocumentに関しての意味合いを同様に有する。アンドゥ操作は、階層内の他の文書に影響する変更を考慮に入れて、例えば、図１７（ｂ）に示されるような連鎖状の階層における全ての文書の間で整合性が維持されることを保証する。
【０１５５】
UndoWrapper７０７は、DocumentContainer２０３内のSubDocumentに関連するアンドゥオブジェクトをラップし、それらをRootDocumentに関連するアンドゥオブジェクトに結合させる。UndoWrapper７０７は、UndoableEditAcceptor（アンドゥアブルエディットアクセプタ：アンドゥ可能編集受付部）７０９に利用可能なアンドゥオブジェクトの収集を実行する。
【０１５６】
UndoManager７０６及びUndoWrapper７０７は、UndoableEditAcceptor７０８及びUndoableEditSource（アンドゥアブルエディットソース）７０８に接続される。当業者には理解

されるように、Document７０５はUndoableEditSource７０８であってもよく、取消可能な編集オブジェクトのソースであってもよい。

【０１５７】

１０．アンドゥコマンド及びアンドゥフレームワーク

図１８（ａ）及び図１８（ｂ）は、アンドゥフレームワーク及びアンドゥコマンドについて更なる詳細を提供する。図１８（ａ）に示されるように、UndoCommand８０１、RedoCommand８０２、及びUndoableEditCommand８０３は、図１１（ｂ）に示したようにCommandInvoker１０５１に積むことができるコマンドであり、順に実行される。UndoableEditCommand８０３は、UndoableEditSource７０８及びUndoableEditAcceptor７０９に更にアタッチされる。「foo」EditCommand８０３及び「bar」EditCommand８０４は、UndoableEditCommandの例である。

【０１５８】

図１８（ｂ）は、UndoableEditCommandの実行を示す。まず、ユーザが編集コマンドを使用してDocument７０５を編集すると仮定する。第１ステップＳ１では、UndoableEditAcceptor７０９が、Document７０５のＤＯＭツリーであるUndoableEditSource７０８にアタッチされる。第２ステップＳ２では、ユーザにより発行されたコマンドに基づいて、Document７０５がＤＯＭのＡＰＩを用いて編集される。第３ステップＳ３では、ミューテーションイベントのリスナーが、変更がなされたことを通知される。すなわち、このステップでは、ＤＯＭツリーの全ての変更を監視するリスナーが編集操作を検知する。第４ステップＳ４では、UndoableEditがUndoManager７０６のオブジェクトとして格納される。第５ステップＳ５では、UndoableEditAcceptor７０９がUndoableEditSource７０８からデタッチされる。UndoableEditSource７０８は、Document７０５自身であってもよい。

【０１５９】

１１．システムへの文書のロードに関する手順

上記のサブセクションでは、システムの様々なコンポーネント及びサブコンポーネントについて説明した。以下、これらのコンポーネントの使用に関する方法論について説明する。図１９は、文書処理／管理システムに文書がロードされる様子の概要を示す。それぞれのステップは、図２４－２８において、特定の例に関連して詳述される。

【０１６０】

簡単には、文書処理／管理システムは、文書に含まれるデータからなるバイナリデータストリームからＤＯＭを生成する。ApexNode（エイペックスノード：頂点ノード）が、Zoneに属する文書の一部の注目対象のために生成される。つづいて、対応するPaneが同定される。同定されたPaneは、ApexNode及び物理的な画面表面からZone及びCanvasを生成する。Zoneは、次に、それぞれのノードにFacetを生成し、それらに必要とされる情報を提供する。Canvasは、ＤＯＭツリーから、ノードをレンダリングするためのデータ構造を生成する。

【０１６１】

より詳細には、図１９（ａ）を参照して、ステップ０では、ＸＨＴＭＬ及びＳＶＧの双方の内容を表現した複合文書がストレージ９０１からロードされる。文書のＤＯＭツリー９０２が生成される。ＤＯＭツリーはApexNode９０５（ＸＨＴＭＬ）を有し、他の枝へ降りていくと、二重線で示される境界に遭遇し、異なるボキャブラリであるＳＶＧのApexNode９０６に続く。このような複合文書の表現は、文書が表現され、最終的に表示のためにレンダリングされる様子を理解するのに有用である。

【０１６２】

次に、文書を保持する、対応するDocumentContainer９０３が生成される。DocumentContainer９０３は、DocumentManager９０４にアタッチされる。ＤＯＭツリーは、ルートノードと、オプションで複数のセカンダリノードを含む。

【０１６３】

一般に、このような文書は、テキスト及びグラフィクスの双方を含む。したがって、ＤＯＭツリーは、例えば、ＸＨＴＭＬサブツリーだけでなくＳＶＧサブツリーを有してもよ

い。ＸＨＴＭＬサブツリーは、ＸＨＴＭＬのApexNode９０５を有する。同様に、ＳＶＧサブツリーは、ＳＶＧのApexNode９０６を有する。

【０１６４】

再び図１９（ａ）を参照して、ステップ１では、ApexNodeが、画面の論理的なレイアウトであるPane９０７にアタッチされる。ステップ２では、Pane９０７は、ApplicationCore９０８に、ApexNodeのためのZoneFactoryを要求する。ステップ３では、ApplicationCore９０８は、ZoneFactoryと、ApexNode９０６のためのCanvasFactoryであるEditletとを返す。

【０１６５】

ステップ４では、Pane９０７がZone９０９を生成する。Zone９０９はPane９０７にアタッチされる。ステップ５では、Zone９０９がそれぞれのノードに対してFacetを生成し、対応するノードにアタッチする。ステップ６では、Pane９０７がCanvas９１０を生成する。Canvas９１０はPane９０７にアタッチされる。Canvas９１０には様々なCommandが含まれる。ステップ７では、Canvas９１０が文書を画面にレンダリングするためのデータ構造を構築する。ＸＨＴＭＬの場合、これはボックスツリー構造を含む。

【０１６６】

図１９（ｂ）は、ＭＶＣパラダイムを用いてZoneの構成の概要を示す。この場合、ZoneFactoryにより生成されるZone及びFacetは文書に関連した入力であるから、モデル（Ｍ）はZone及びFacetを含む。Canvasと、Editletを用いて文書を画面にレンダリングするためのデータ構造は、ユーザが画面上に見る出力であるから、ビュー（Ｖ）はCanvas及びデータ構造に対応する。Commandは、文書とその様々な関係に対して制御操作を実行するので、コントロール（Ｃ）はCanvasに含まれるCommandを含む。

【０１６７】

１２．文書の表現

図２０を用いて、複合文書及びその様々な表現の例について以下に説明する。この例で使用される文書は、テキストと画像の双方を含む。テキストは、ＸＨＴＭＬを用いて表され、画像は、ＳＶＧを用いて表される。図２０は、文書の構成要素及び対応するオブジェクトの関係のＭＶＣ表現を詳細に示す。この例において、Document１００１は、Document１００１を保持するDocumentContainer１００２にアタッチされる。文書はＤＯＭツリー１００３により表現される。ＤＯＭツリーは、図１９（ａ）を参照して上述した対応するFacetを有するApexNode１００４及び子孫の他のノードを含む。

【０１６８】

ApexNodeは、黒丸で表される。頂点でないノードは、白丸で表される。ノードを編集するために用いられるFacetは、三角形で表され、対応するノードにアタッチされる。文書がテキストと画像を有するので、この文書のＤＯＭツリーは、ＸＨＴＭＬ部分とＳＶＧ部分を含む。ApexNode１００４は、ＸＨＴＭＬサブツリーの最上のノードである。これは、文書のＸＨＴＭＬ部分の物理的な表現のための最上PaneであるXHTMLPane１００５にアタッチされる。ApexNodeは、文書１００１のＤＯＭツリーの一部であるXHTMLZone１００６にもアタッチされる。

【０１６９】

Node１００４に対応するFacet１０４１も、XHTMLZone１００６にアタッチされる。XHTMLZone１００６は、XHTMLPane１００５にアタッチされる。XHTMLEditletは、文書の論理的な表現であるXHTMLCanvas１００７を生成する。XHTMLCanvas１００７は、XHTMLPane１００５にアタッチされる。XHTMLCanvas１００７は、Document１００１のＸＨＴＭＬ要素のためのBoxTree１００９を生成する。ボックスツリーは、図示されるように、「html」ボックス、「body」ボックス、「head」ボックス、及び／又は「table」ボックスの適切な結合により表現される。文書のＸＨＴＭＬ部分を保持し描画するために必要な様々なCommand１００８も、XHTMLCanvas１００５に追加される。

【０１７０】

同様に、文書のＳＶＧサブツリーのApexNode１０１０は、文書のＳＶＧ要素を表現する

Document１００１のＤＯＭツリーの一部であるSVGZone１０１１にアタッチされる。ApexNode１０１０は、文書のＳＶＧ部分の物理的な表現の最上のPaneであるSVGPane１０１３にアタッチされる。文書のＳＶＧ部分の論理的な表現を表すSVGCanvas１０１２は、SVGEditletにより生成され、SVGPane１０１３にアタッチされる。画面上に文書のＳＶＧ部分をレンダリングするためのデータ構造及びコマンド１０１４は、SVGCanvasにアタッチされる。例えば、このデータ構造は、図示されるように、円、線、長方形などを含んでもよい。

【０１７１】

図２０に関連して説明された文書例の表現の一部について、図２１（ａ）（ｂ）に関連して、前述したＭＶＣパラダイムを用いて更に説明する。図２１（ａ）は、文書１００１のＸＨＴＭＬ要素におけるＭＶの関係を簡略化して示す。モデルは、Document１００１のＸＨＴＭＬ要素のためのXHTMLZone１１０３である。XHTMLZoneのツリーには、いくつかのNode及びそれらに対応するFacetが含まれる。対応するXHTMLZone及びPaneは、ＭＶＣパラダイムのモデル（Ｍ）部分の一部である。ＭＶＣパラダイムのビュー（Ｖ）部分は、Document１００１のＸＨＴＭＬコンポーネントの、対応するXHTMLCanvas１１０２及びBoxTreeである。文書のＸＨＴＭＬ部分は、Canvasと、それに含まれるCommandを使用して画面に描写される。キーボードやマウス入力などのイベントは、図示されるように、逆方向へ進む。

【０１７２】

SourcePaneは、更なる機能、すなわち、ＤＯＭの保有者としての役割を有する。図２１（ｂ）は、図２１（ａ）に示したDocument１００１の要素に対するボキャブラリコネクションを提供する。ＤＯＭホルダーとして機能するSourcePane１１０３は、文書のソースＤＯＭツリーを含む。ConnectorTree１１０４は、ConnectorFactoryにより生成され、デスティネーションＤＯＭの保有者としても機能するDestinationPane１１０５を生成する。DestinationPane１１０５は、XHTMLDestinationCanvas１１０６としてボックスツリーの形式でレイアウトされる。

【０１７３】

１３．プラグインサブシステム、ボキャブラリコネクション、及びコネクタの関係

図２２（ａ）－（ｃ）は、それぞれ、プラグインサブシステム、ボキャブラリコネクション、及びConnectorに関連する更なる詳細を示す。プラグインサブシステムは、文書処理／管理システムに機能を追加又は交換するために用いられる。プラグインサブシステムは、ServiceBroker１０４１を含む。図２２（ａ）に示すように、「私自身のＸＭＬボキャブラリ」のＶＣＤファイルがＶＣのプラグインに結合される。このプラグインは、「ＭｙＯｗｎＸＭＬ」のConnectorFactoryTree及びVocabulary（ZoneFactory、Editlet）を含む。ServiceBroker１０４１にアタッチされるZoneFactoryService１２０１は、文書の部分のZoneを生成する。EditletService１２０２も、ServiceBrokerにアタッチされる。EditletService１２０２は、Zone中のNodeに対応するCanvasを生成する。

【０１７４】

ZoneFactoryの例は、XHTMLZone及びSVGZoneをそれぞれ生成するXHTMLZoneFactory１２１１及びSVGZoneFactory１２１２である。文書例に関連して前述したように、文書のテキスト要素は、XHTMLZoneを生成することにより表現されてもよいし、画像はSVGZoneを用いて表現されてもよい。EditletServiceの例は、XHTMLEditlet１２２１及びSVGEditlet１２２２を含む。

【０１７５】

図２２（ｂ）は、ボキャブラリコネクションに関連する更なる詳細を示す。ボキャブラリコネクションは、前述したように、文書処理／管理システムの重要な特徴であり、２つの異なる方法で文書の整合のとれた表現及び表示を可能とする。ConnectorFactory３０３を保持するVCManager３０２は、ボキャブラリコネクションサブシステムの一部であり、ボキャブラリコネクション記述子を受け取るためにＶＣＤに結合され、VCCommand３０１を生成する。図２２（ｃ）に示すように、ConnectorFactory３０３は、文書のConnector３０４を生成する。前述したように、Connectorは、ソースＤＯＭ中のノードを監視し、

２つの表現の間の整合性を維持するために、デスティネーションＤＯＭ中のノードを修正する。

【０１７６】

Templateは、いくつかのノードの変換ルールを表す。ボキャブラリコネクション記述子（ＶＣＤ）ファイルは、特定のパス又はルールを満たす要素又は要素の集合を他の要素に変換するいつくかのルールを表すTemplateのリストである。VocabularyTemplate３０５及びCommandTemplate３１３１は、全てVCManager３０２にアタッチされる。VCManagerは、ＶＣＤファイル中の全てのセクションを管理するオブジェクトである。１つのＶＣＤファイルに対して、１つのVCManagerオブジェクトが生成される。

【０１７７】

図２２（ｃ）は、Connectorに関連する更なる詳細を提供する。ConnectorFactory３０３は、ソース文書からConnectorを生成する。ConnectorFactory３０３は、Vocabulary、Template、及びElementTemplateにアタッチされ、それぞれ、VocabularyConnector、TemplateConnector、ElementConnectorを生成する。

【０１７８】

VCManager３０２は、ConnectorFactory３０３を保持する。Vocabularyを生成するために、対応するＶＣＤファイルが読み込まれる。こうして、ConnectorFactory３０３が生成される。このConnectorFactory３０３は、Zoneを生成するZoneFactory２０５及びCanvasを生成するEditletService２０６に関連する。

【０１７９】

つづいて、ターゲットボキャブラリのEditletServiceが、VCCanvasを生成する。VCCanvasは、デスティネーションＤＯＭツリーのノードを生成する。VCCanvasは、ソースＤＯＭツリー又はZoneにおけるApexNodeのConnectorも生成する。必要に応じて、子のConnectorが再帰的に生成される。ConnectorTreeは、ＶＣＤファイル中のテンプレートの集合により生成される。

【０１８０】

テンプレートは、マークアップ言語の要素を他の要素に変換するためのルールの集合である。例えば、各テンプレートは、ソースＤＯＭツリー又はZoneにマッチされる。適切にマッチした場合には、頂点Connectorが生成される。例えば、テンプレート「A/*/D」は、間にどんなノードがあるかに関係なく、ノードＡで始まりノードＤで終わる全ての枝に合致する。同様に、「//B」は、ルートからの全ての「B」ノードに一致する。

【０１８１】

１４．ConnectorTreeに関係するＶＣＤファイルの例

特定の文書と関係する処理を説明する例を続ける。「MySampleXML」というタイトルの文書が文書処理／管理システムにロードされる。図２３は、「MySampleXML」ファイルのための、VCManager及びConnectorFactoryTreeを用いたＶＣＤスクリプトの例を示す。スクリプトファイル中のボキャブラリセクション、テンプレートセクションと、VCManagerにおける対応するコンポーネントが示される。タグ「vcd:vocabulary」において、属性「match」は「sample:root」、「label」は「MySampleXML」、「call-template」は「sample template」となっている。

【０１８２】

この例では、Vocabularyは、「MySampleXML」のVCManagerにおいて「sample:root」として頂点要素を含む。対応するＵＩラベルは、「MySampleXML」である。テンプレートセクションにおいて、タグは「vcd:template」であり、名前は「sample:template」である。

【０１８３】

１５．ファイルがシステムにロードされる方法の詳細な例

図２４－２８は、文書「MySampleXML」のロードについての詳細な記述を示す。図２４（ａ）に示されるステップ１では、文書がストレージ１４０５からロードされる。DOMServiceは、ＤＯＭツリー及びDocumentManager１４０６と対応するDocumentContainer１４０

１を生成する。DocumentContainerは、DocumentManager１４０６にアタッチされる。文書は、ＸＨＴＭＬ及びMySampleXMLのサブツリーを含む。ＸＨＴＭＬのApexNode１４０３は、タグ「xhtml:html」が付されたＸＨＴＭＬの最上のノードである。「MySampleXML」のApexNode１４０４は、タグ「sample:root」が付された「MySampleXML」の最上ノードである。

【０１８４】

図２４（ｂ）に示されるステップ２では、RootPaneが文書のXHTMLZone、Facet、及びCanvasを生成する。Pane１４０７、XHTMLZone１４０８、XHTMLCanvas１４０９、及びBoxTree１４１０が、ApexNode１４０３、及び、ステップ１－５において、図中に示される関係に応じてFacetが関連づけられる他のノードに対応して生成される。

【０１８５】

図２４（ｃ）に示されるステップ３では、XHTMLZoneが知らないタグ「sample:root」を発見し、XHTMLCanvasの領域からSubPaneを生成する。

【０１８６】

図２５に示されるステップ４では、SubPaneが「sample:root」を扱うことができ、適切なZoneを生成可能なZoneFactoryを得る。このZoneFactoryは、ZoneFactoryを実行可能なVocabulary内にある。それは、「MySampleXML」のVocabularySectionの内容を含む。

【０１８７】

図２６に示されるステップ５では、「MySampleXML」に対応し、VCManagerに関連するVocabularyが、DefaultZone１６０１を生成する。対応するEditletが生成され、対応するCanvasを生成するためにSubPane１５０１が提供される。Editletは、VCCanvasを生成する。そして、それはTemplateSectionを呼ぶ。ConnectorFactoryTreeも含まれている。ConnectorFactoryTreeは、VCCanvasの一部を形成するConnectorTreeとなる全てのConnectorを生成する。ルートペインとＸＨＴＭＬゾーンの関係や、ＸＨＴＭＬキャンバスと文書のＸＨＴＭＬ部分に関連する頂点ノードに対するボックスツリーとの関係は、上記の議論から明確である。

【０１８８】

図２７には、ソースＤＯＭツリー、ＶＣキャンバス、及びデスティネーションＤＯＭツリーの間の既に説明した対応に基づいて、ステップ６が示される。ステップ６では、各ConnectorがデスティネーションＤＯＭオブジェクトを生成する。コネクタのうちのいくつかはxpath情報を含んでいる。xpath情報は、変更／修正を監視する必要のあるソースＤＯＭツリーの部分集合を決定するために使用される１以上のxpath表現を含む。

【０１８９】

図２８には、ソース、ＶＣ及びデスティネーションの関係にしたがって、ステップ７が示される。ステップ７では、ボキャブラリは、ソースＤＯＭのペインからデスティネーションＤＯＭツリーのDestinationPaneを作成する。これは、SourcePaneに基づいてなされる。デスティネーションツリーのApexNodeは、DestinationPane及び対応するZoneにアタッチされる。DestinationPaneは、DestinationCanvasを生成し、文書をデスティネーションのフォーマットでレンダリングするためのデータ構造及びコマンドを構築する、自身のEditletを提供される。

【０１９０】

図２９（ａ）は、対応するソースノードを持たず、デスティネーションツリーにのみ存在するノード上でイベントが発生したときのフローを示す。第１ステップにおいて、マウスイベント、キーボードイベントなど、Canvasが取得したイベントは、デスティネーションツリーを通過して、ElementTemplateConnectorに伝達される。ElementTemplateConnectorは対応するソースノードを持たないので、伝達されたイベントはソースノードに対する編集操作ではない。ElementTemplateConnectorは、伝達されたイベントがCommandTemplateに記述されたコマンドに合致すれば、第２及び第３ステップにおいて、それに対応するActionを実行する。合致するコマンドがなければ、ElementTemplateConnectorは、伝達されたイベントを無視する。

【０１９１】
図２９（ｂ）は、TextOfConnectorによりソースノードに対応づけられているデスティネーションツリーのノード上でイベントが発生したときのフローを示す。TextOfConnectorは、ソースＤＯＭツリーのＸＰａｔｈで指定されたノードからテキストノードを取得して、デスティネーションＤＯＭツリーのノードにマッピングする。第１ステップにおいて、マウスイベント、キーボードイベントなど、Canvasが取得したイベントは、デスティネーションツリーを通過して、TextOfConnectorに伝達される。TextOfConnectorは、伝達されたイベントを、対応するソースノードの編集コマンドにマッピングし、Queue１０５３に積む。編集コマンドは、Facetを介して実行されるＤＯＭのＡＰＩコールの集合である。第２ステップにおいて、キューに積まれたコマンドが実行されると、ソースノードが編集される。ソースノードが編集されると、第３ステップにおいて、ミューテーションイベントが発行され、リスナーとして登録されたTextOfConnectorにソースノードの変更が通知される。TextOfConnectorは、ソースノードの変更を、対応するデスティネーションノードに反映させるように、デスティネーションツリーを再構築する。このとき、TextOfConnectorを含むテンプレートに、「for each」や「for loop」などの制御文が含まれている場合、ConnectorFactoryがこの制御文を再評価し、TextOfConnectorを再構築した後、デスティネーションツリーが再構築される。
【０１９２】
以上、本発明を実施の形態をもとに説明した。この実施の形態は例示であり、それらの各構成要素や各処理プロセスの組合せにいろいろな変形例が可能なこと、またそうした変形例も本発明の範囲にあることは当業者に理解されるところである。
【０１９３】
実施の形態では、ＸＭＬ文書を処理する例について説明したが、本実施の形態の文書処理装置１００は、他のマークアップ言語、例えば、ＳＧＭＬ、ＨＴＭＬなどで記述された文書も同様に処理可能である。
【図面の簡単な説明】
【０１９４】
【図１】本発明の実施の形態に係る文書処理装置の構成を示すブロック図である。
【図２】ＸＭＬ文書の例を示す図である。
【図３】図２に示したＸＭＬ文書をＨＴＭＬで記述された表にマッピングする例を示す図である。
【図４（ａ）】図２に示したＸＭＬ文書を図３に示した表にマッピングするための定義ファイルの例を示す図である。
【図４（ｂ）】図２に示したＸＭＬ文書を図３に示した表にマッピングするための定義ファイルの例を示す図である。
【図５】図２に示したＸＭＬ文書を、図３に示した対応によりＨＴＭＬにマッピングして表示した画面の例を示す図である。
【図６】本発明で使用可能なグラフィカルユーザインターフェースを示す図である。
【図７】本発明により生成された画面レイアウトの他の例を示す図である。
【図８】本発明によるＸＭＬ文書の編集画面の一例を示す図である。
【図９】本発明により編集されるＸＭＬ文書の他の例を示す図である。
【図１０】本発明で使用可能な編集画面を示す図である。
【図１１（ａ）】開示された文書処理管理システムの実施の形態の基本として機能する要素の構成を示す図である。
【図１１（ｂ）】文書処理管理システム全体のブロック図を示す図である。
【図１１（ｃ）】文書処理管理システム全体のブロック図を示す図である。
【図１２】文書管理部の実施の形態の更なる詳細を示す図である。
【図１３】ボキャブラリコネクションサブシステムの実施の形態の更なる詳細を示す図である。
【図１４】図１４（ａ）は、プログラム起動部及びその他の構成の関係の実施の形態の更

なる詳細を示す図であり、図１４（ｂ）は、サービス仲介部及びその他の構成との関係の実施の形態の更なる詳細を示す図であり、図１４（ｃ）は、サービスの実施の形態の更なる詳細を示す図であり、図１４（ｄ）は、サービスの例を示す図であり、図１４（ｅ）は、プログラム起動部とユーザアプリケーションとの間の関係の更なる詳細を示す図である。

【図１５】図１５（ａ）は、プログラム起動部によりロードされたアプリケーションサービスの構造の更なる詳細を示す図であり、図１５（ｂ）は、フレーム、メニューバー、及びステータスバーの間の関係の例を示す図である。

【図１６】図１６（ａ）は、アプリケーションコアの実施の形態に関連する更なる詳細を示す図であり、図１６（ｂ）は、スナップショットの実施の形態に関連する更なる詳細を示す図である。

【図１７】図１７（ａ）は、文書管理部の実施の形態に関連する更なる詳細を示す図であり、図１７（ｂ）は、右側に、一連の文書Ａ－Ｅが階層的に配置される様子を示し、左側に、右側に示された文書の階層が画面に表示される様子の例を示した図である。

【図１８】図１８（ａ）（ｂ）は、アンドゥフレームワークとアンドゥコマンドの実施の形態の更なる詳細を示す図である。

【図１９】図１９（ａ）は、図１１（ｂ）（ｃ）に示された文書処理管理システムにおいて文書がロードされる様子を示す図であり、図１９（ｂ）は、ＭＶＣパラダイムを用いてゾーンの構造の概略を示す図である。

【図２０】本発明による文書とその表現の例を示す図である。

【図２１】図２１（ａ）は、図２０に示された文書のＸＨＴＭＬ要素に関するＭＶの関係を簡素化した図であり、図２１（ｂ）は、図２１（ａ）に示された文書に関するボキャブラリコネクションを示す図である。

【図２２】図２２（ａ）－（ｃ）は、プラグインサブシステム、ボキャブラリコネクション、及びコネクタのそれぞれの実施の形態に関連する更なる詳細を示す図である。

【図２３】ＭｙＳａｍｐｌｅＸＭＬファイルに対するボキャブラリコネクションマネージャ及びコネクタファクトリツリーを用いたＶＣＤスクリプトの例を示す図である。

【図２４】図２４（ａ）－（ｃ）は、ＭｙＳａｍｐｌｅＸＭＬ文書の例を図１１（ｂ）の文書処理管理システムの例にロードする手順０－３を示す図である。

【図２５】ＭｙＳａｍｐｌｅＸＭＬ文書の例を図１１（ｂ）の文書処理管理システムの例にロードする手順４を示す図である。

【図２６】ＭｙＳａｍｐｌｅＸＭＬ文書の例を図１１（ｂ）の文書処理管理システムの例にロードする手順５を示す図である。

【図２７】ＭｙＳａｍｐｌｅＸＭＬ文書の例を図１１（ｂ）の文書処理管理システムの例にロードする手順６を示す図である。

【図２８】ＭｙＳａｍｐｌｅＸＭＬ文書の例を図１１（ｂ）の文書処理管理システムの例にロードする手順７を示す図である。

【図２９】図２９（ａ）は、対応するソースノードを持たず、デスティネーションツリーのみに依存するノードに発生したイベントの流れを示す図であり、図２９（ｂ）は、テキストオブコネクタによりソースノードに関連づけられたデスティネーションノードに発生したイベントの流れを示す図である。

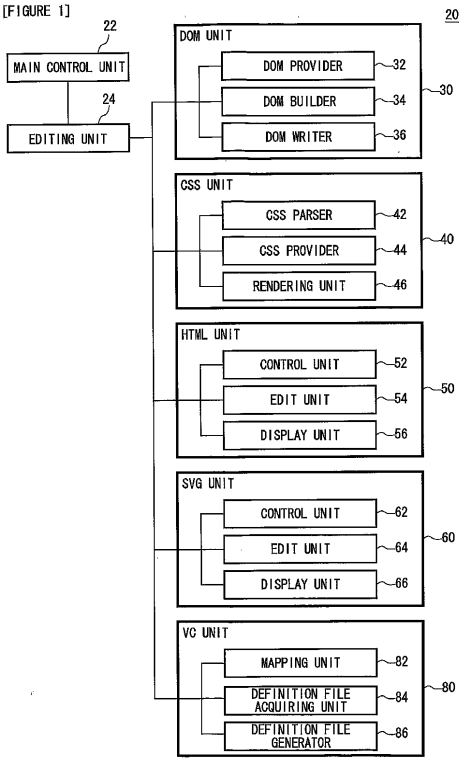
【符号の説明】

【０１９５】

２０　文書処理装置、２２　主制御ユニット、２４　編集ユニット、３０　ＤＯＭユニット、３２　ＤＯＭ提供部、３４　ＤＯＭ生成部、３６　出力部、４０　ＣＳＳユニット、４２　解析部、４４　ＣＳＳ提供部、４６　レンダリング部、５０　ＨＴＭＬユニット、５２，６２　制御部、５４，６４　編集部、５６，６６　表示部、６０　ＳＶＧユニット、８０　ＶＣユニット、８２　マッピング部、８４　定義ファイル取得部、８６　定義ファイル生成部。

【図１】

[FIGURE 1]



【図２】

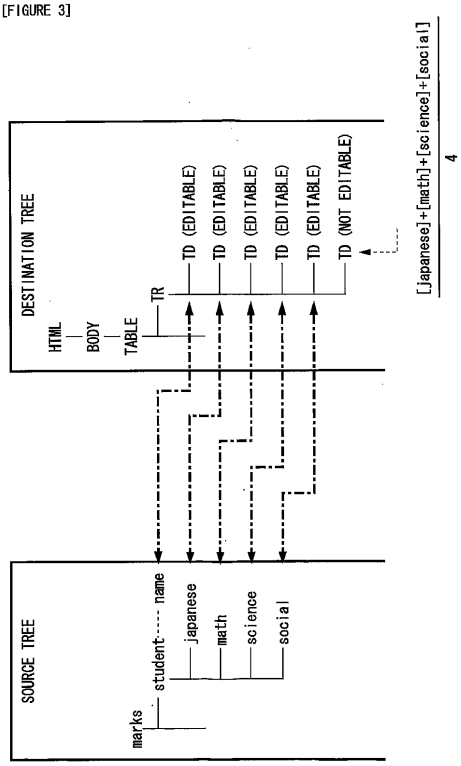
[FIGURE 2]

```
<?xml version="1.0" ?>

<?org.chimaira vocabulary-connection href="records.vcd" ?>
<marks xmlns="http://xmlns.justsystem.com/sample/records">
  <student name="A">
    <japanese>90</japanese>
    <math>50</math>
    <science>75</science>
    <social>60</social>
  </student>
  <student name="B">
    <japanese>45</japanese>
    <math>60</math>
    <science>55</science>
    <social>50</social>
  </student>
  <student name="C">
    <japanese>55</japanese>
    <math>45</math>
    <science>95</science>
    <social>40</social>
  </student>
  <student name="D">
    <japanese>25</japanese>
    <math>35</math>
    <science>40</science>
    <social>15</social>
  </student>
</marks>
```

【図３】

[FIGURE 3]



【図４】

[FIGURE 4]

```
<?xml version="1.0"?>

<vc:vcd xmlns:vc="http://xmlns.chimaira.org/vcd"
        xmlns:src="http://xmlns.justsystem.com/sample/records"
        xmlns="http://www.w3.org/1999/xhtml"
        version="1.0">

<!-- Commands -->
<vc:command name="add student">
  <vc:insert-fragment
      target="ancestor-or-self::src:student"
      position="after">
    <src:student/>
  </vc:insert-fragment>
</vc:command>
<vc:command name="delete student">
  <vc:delete-fragment target="ancestor-or-self::src:student" />
</vc:command>

<!-- Templates -->
<vc:vc-template match="src:marks" name="report card" >

  <vc:ui command="add student">
    <vc:mount-point>
      /MenuBar/report card/add student
    </vc:mount-point>
  </vc:ui>
  <vc:ui command="delete student">
    <vc:mount-point>
      /MenuBar/report card/delete student
    </vc:mount-point>
  </vc:ui>

  <html>
    <head>
      <title>report card</title>
      <style>
        td,th {
          text-align:center;
          border-right:solid black 1px;
          border-bottom:solid black 1px;
          border-top:none 0px;
          border-left:none 0px;
        }
        table{
          border-top:solid black 2px;
          border-left:solid black 2px;
          border-right:solid black 1px;
          border-bottom:solid black 1px;
          border-spacing:0px;
        }
```

【図４】

Continuation of Figure 4

```
        tr{
          border:none;
        }
        .data{
          padding:0.2em 0.5em;
        }
      </style>
    </head>
    <body>
      <h1>MARKS TABLE</h1>
      <table>
        <tr><th><div class="data">NAME</div></th>
        <th></th>
        <th><div class="data">JAPN</div></th>
        <th><div class="data">MATH</div></th>
        <th><div class="data">SCI</div></th>
        <th><div class="data">SS</div></th>
        <th></th>
        <th><div class="data">AVE</div></th></tr>
        <vc:apply-templates select="src:student" />
      </table>
    </body>
  </html>
</vc:vc-template>

<vc:template match="src:student">
  <tr>
    <td><div class="data">
      <vc:text-of select="@name" fallback="no name"/></div></td>
    <td></td>
    <td><div class="data">
      <vc:text-of select="src:japanese"
          fallback="0" type="vc:integer" /></div></td>
    <td><div class="data">
      <vc:text-of select="src:math"
          fallback="0" type="vc:integer" /></div></td>
    <td><div class="data">
      <vc:text-of select="src:science"
          fallback="0" type="vc:integer" /></div></td>
    <td><div class="data">
      <vc:text-of select="src:social"
          fallback="0" type="vc:integer" /></div></td>
    <td></td>
    <td><div class="data">
      <vc:value-of
select="(src:japanese+src:math+src:science+src:social) div 4" />
    </div></td>
  </tr>
</vc:template>

</vc:vcd>
```

【図５】

[FIGURE 5]

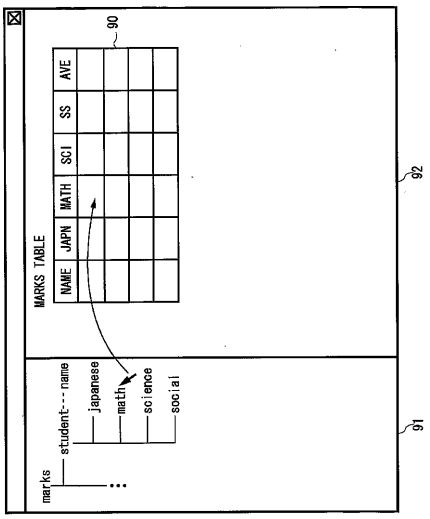
sample.xml

MARKS TABLE 90

NAME	JAPN	MATH	SCI	SS	AVE
A	90	50	75	60	68.8
B	45	60	55	50	52.5
C	55	45	95	40	58.8
D	25	35	40	15	28.8

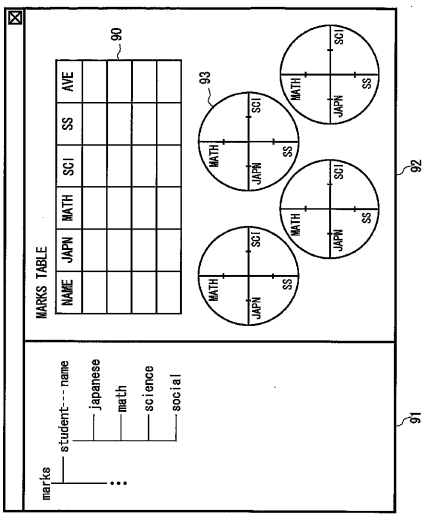
【図６】

[FIGURE 6]



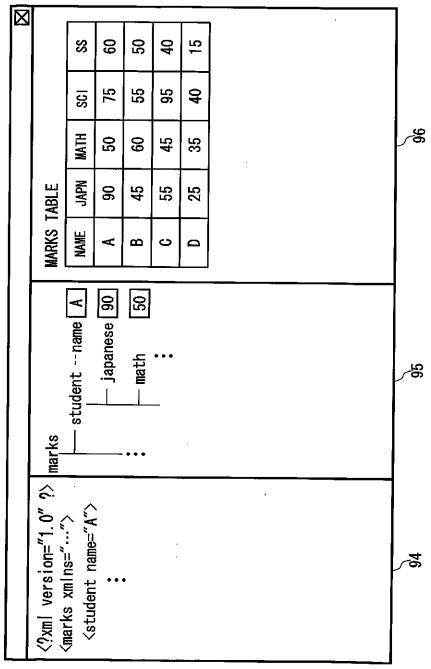
【図７】

[FIGURE 7]



【図８】

[FIGURE 8]



【図９】

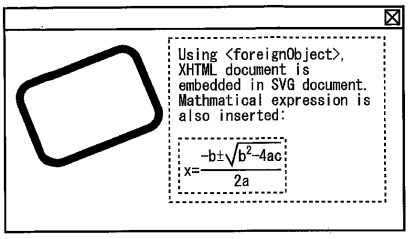
[FIGURE 9]

```
<?xml version="1.0" ?>

<svg xmlns="http://www.w3.org/2000/svg"
     width="400" height="200"
     viewBox="0 0 400 200"
>
  <rect x="-15" y="65" width="150" height="100" rx="20"
        transform="rotate(-20)"
        style="fill:none; stroke:purple; stroke-width:10"
  />
  <foreignObject x="190" y="10" width="200" height="200">
    <html xmlns="http://www.w3.org/1999/xhtml">
      <head><title /></head>
      <body bgcolor="#FFFFCC" text="darkgreen">
        <div style="font-size:12pt">
          Using &lt;foreignObject&gt;, XHTML document is
          embedded in SVG document.
          Mathmatical expression is also inserted:
          <div>
            <math xmlns="http://www.w3.org/1998/Math/MathML">
              <mi>x</mi>
              <mo>=</mo>
              <mfrac>
                <mrow>
                  <mo>−</mo>
                  <mi>b</mi>
                  <mo>±</mo>
                  <msqrt>
                    <mrow>
                      <msup>
                        <mi>b</mi>
                        <mn>2</mn>
                      </msup>
                      <mo>−</mo>
                      <mn>4</mn>
                      <mi>a</mi>
                      <mi>c</mi>
                    </mrow>
                  </msqrt>
                </mrow>
                <mrow>
                  <mn>2</mn>
                  <mi>a</mi>
                </mrow>
              </mfrac>
            </math>
          </div><!— math —>
        </div>
      </body>
    </html>
  </foreignObject>
</svg>
```

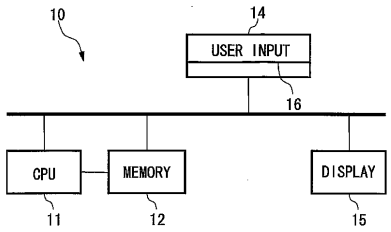
【図１０】

[FIGURE 10]



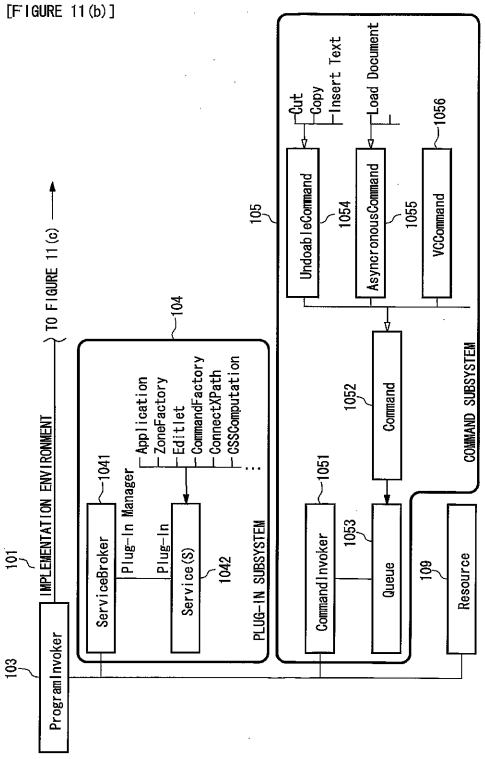
【図１１（ａ）】

[FIGURE 11(a)]



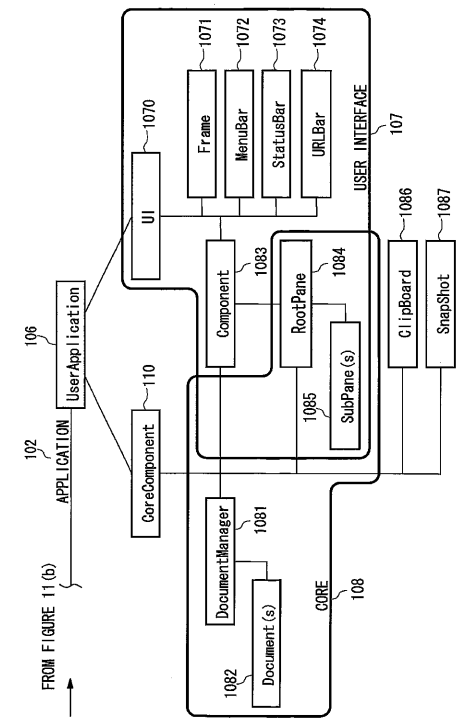
【図１１（ｂ）】

[FIGURE 11(b)]



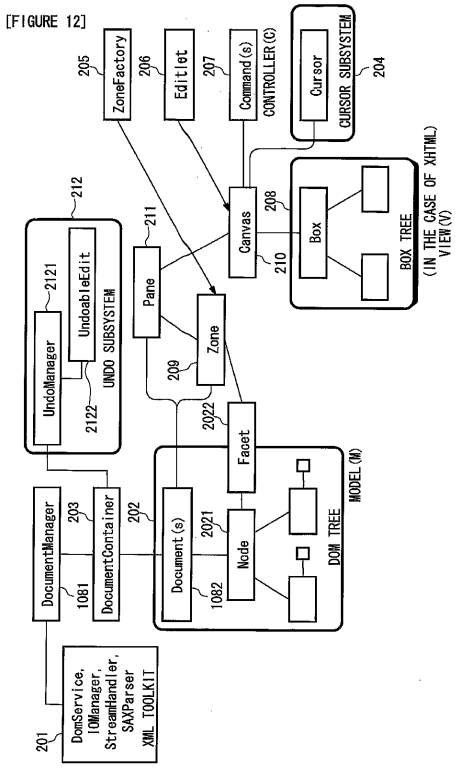
【図１１（ｃ）】

[FIGURE 11(c)]



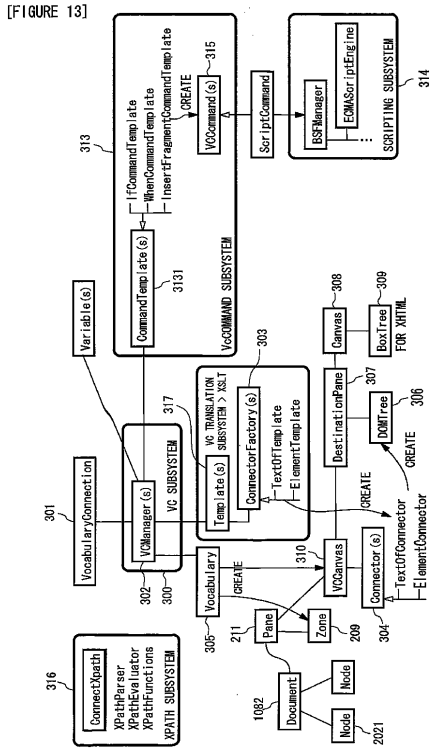
【図１２】

[FIGURE 12]



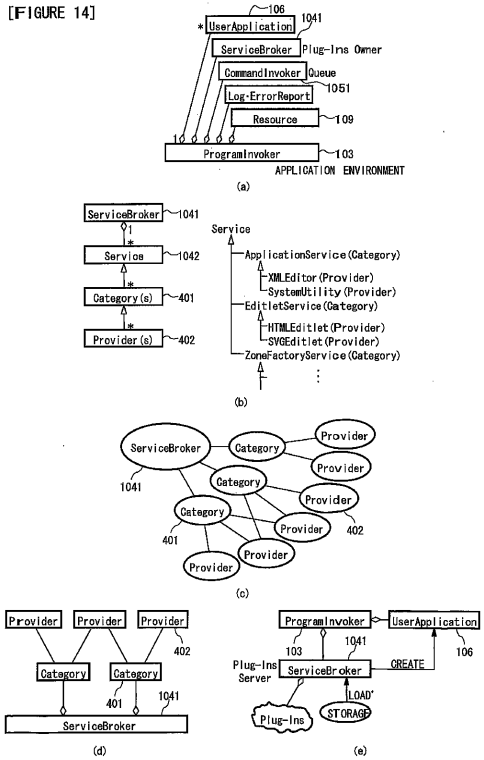
【図１３】

[FIGURE 13]

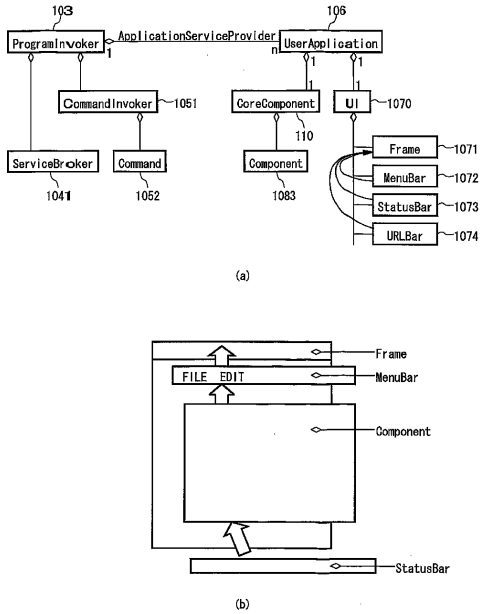


【図１４】

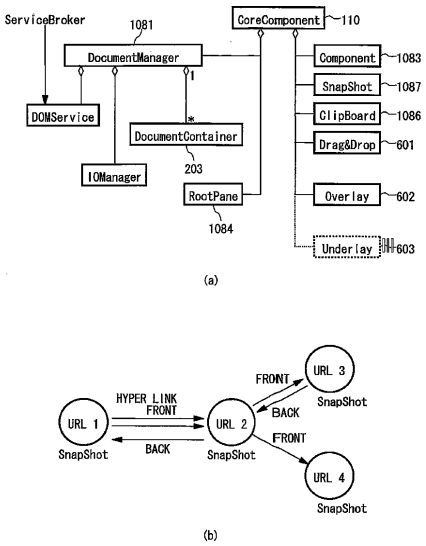
[FIGURE 14]



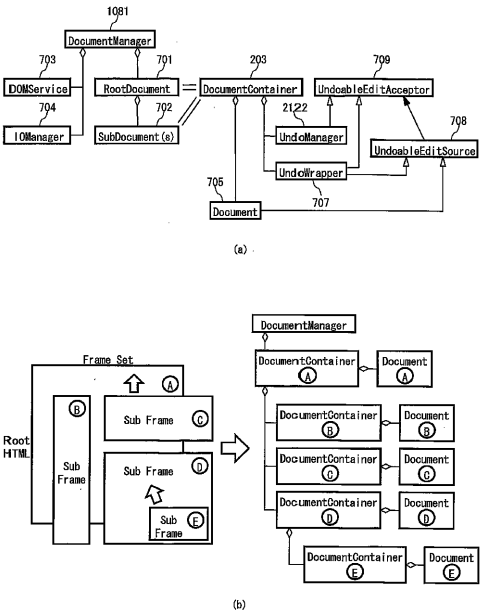
【図１５】
[FIGURE 15]



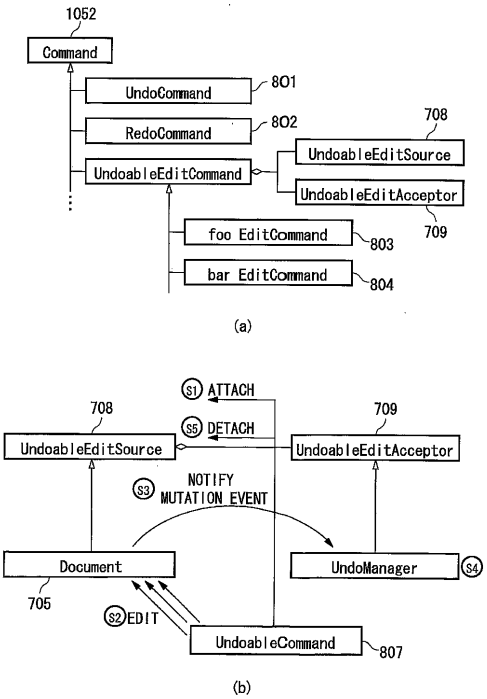
【図１６】
[FIGURE 16]



【図１７】
[FIGURE 17]

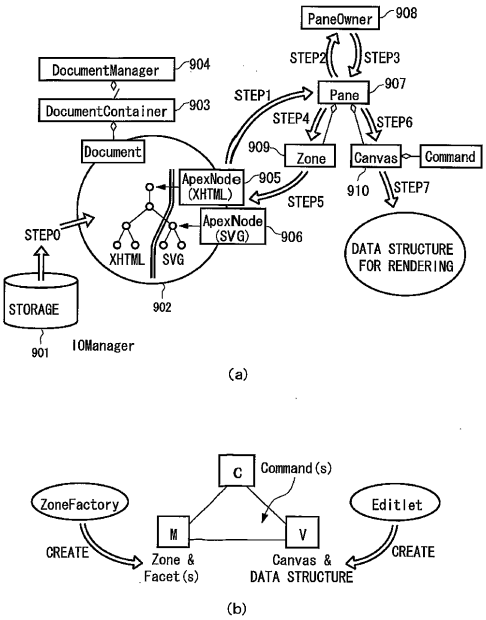


【図１８】
[FIGURE 18]



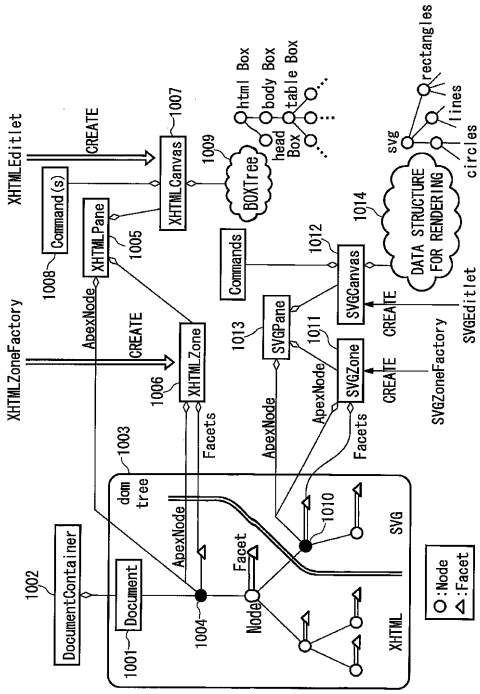
【図１９】

[FIGURE 19]



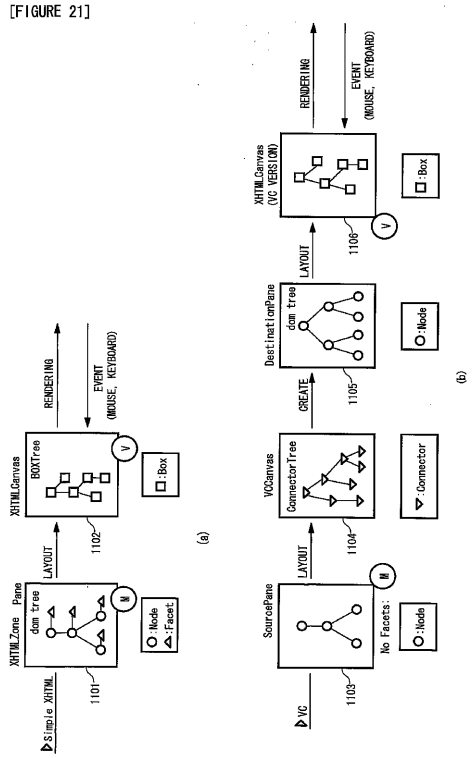
【図２０】

[FIGURE 20]



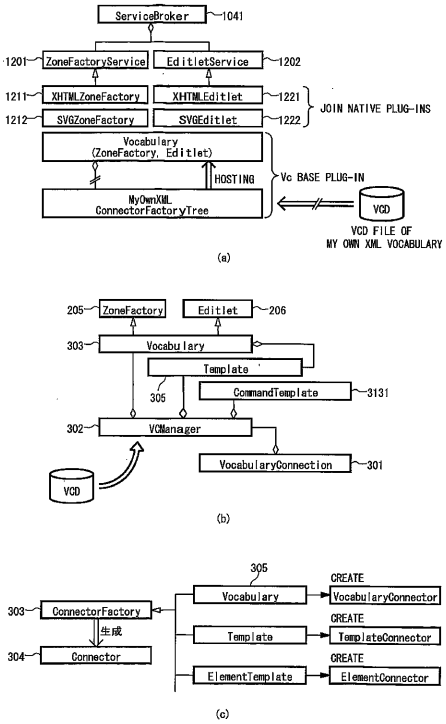
【図２１】

[FIGURE 21]



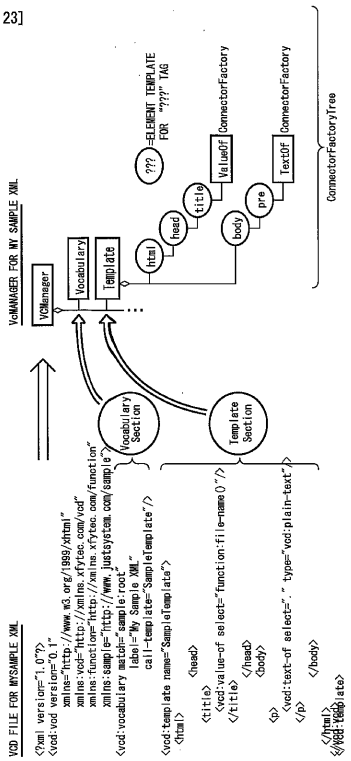
【図２２】

[FIGURE 22]



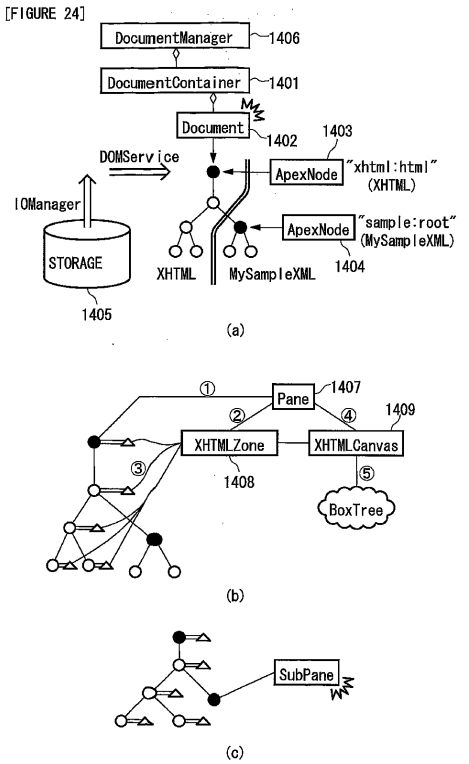
【図２３】

[FIGURE 23]



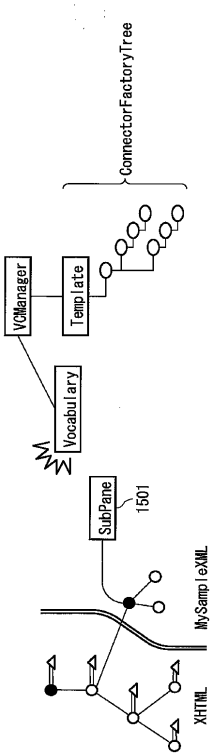
【図２４】

[FIGURE 24]



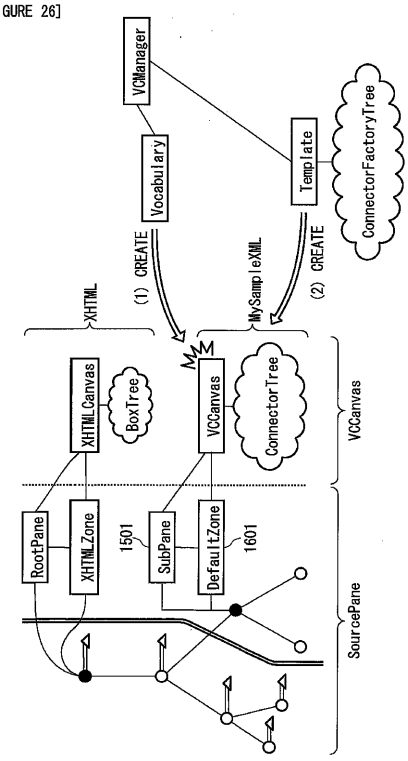
【図２５】

[FIGURE 25]



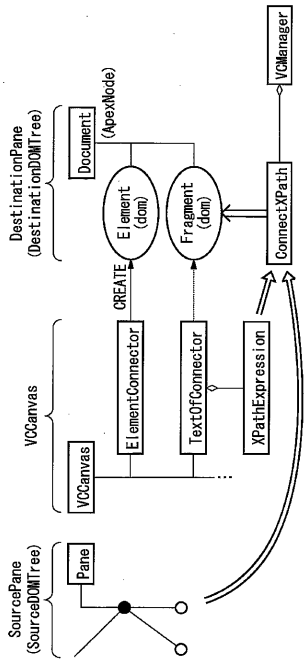
【図２６】

[FIGURE 26]



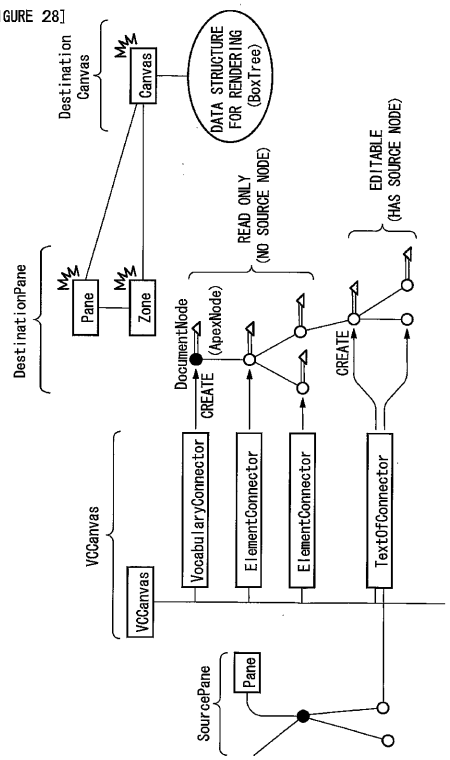
【図２７】

[FIGURE 27]



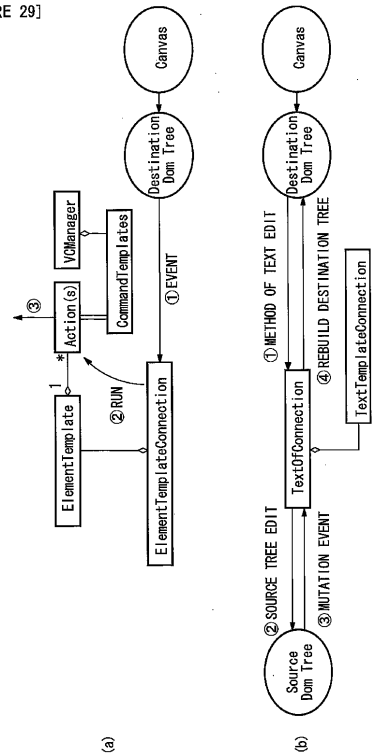
【図２８】

[FIGURE 28]



【図２９】

[FIGURE 29]



【国際調査報告】

INTERNATIONALSEARCHREPORT

International application No.
PCT/JP2005/007287

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl.[7] G06F17/21

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl.[7] G06F17/21

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Published examined utility model applications of Japan 1922-1996
Published unexamined utility model applications of Japan 1971-2005
Registered utility model specifications of Japan 1996-2005
Published registered utility model applications of Japan 1994-2005

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	JP 2004-30582 A (TOSHIBA CORP.) 2004.01.29, all texts, all figures, &US 2004/44965 A1	1-3,6-14,18-30,34-44,46
Y		5,45
A		4,15-17,31-33
X	JP 2004-501450 A (MICROSOFT CORP) 2004.01.15, all texts, all figures, &WO 2001-098927 A &EP 1350180 A2 &AU 200163119 A &BR 200111797 A	1-3,6-14,18-30,34-44,46
Y		5,45
A		4,15-17,31-33

☑ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
08.08.2005	30.08.2005
Name and mailing address of the ISA/JP **Japan Patent Office** 3-4-3, Kasumigaseki, Chiyoda-ku, Tokyo 100-8915, Japan	Authorized officer Hiroyuki Naruse 5M 9192 Telephone No. +81-3-3581-1101 Ext. 3599

Form PCT/ISA/210 (second sheet) (January 2004)

INTERNATIONALSEARCHREPORT

International application No.
PCT/JP2005/007287

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	JP 11-259460 A (TOSHIBA CORP.) 1999.09.24, all texts, all figures, (no family)	5,45

Form PCT/ISA/210 (continuation of second sheet) (January 2004)

INTERNATIONALSEARCHREPORT

International application No.
PCT/JP2005/007287

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:
See extra sheets.

1. ☑ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.
2. ☐ As all searchable claims could be searched without effort justifying an additional fee, this Authority did not invite payment of any additional fee.
3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest.

☑ No protest accompanied the payment of additional search fees.

Form PCT/ISA/210 (continuation of first sheet (2)) (January 2004)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2005/007287

The feature common to both of claims 1, 2 and claims 3-5 (or claims 6-8, claims 9-22) is a matter indicated by Claim 1.

However, the search has revealed that this matter is not novel since it is disclosed in document JP 2004-30582 A.

Consequently the common feature is not a special technical feature within the meaning of PCT Rule 13.2, second sentence, since it makes no contribution over the prior art.

Therefore, there is no feature common to all the claims. Since there is no other common feature which can be considered as a special technical feature within the meaning of PCT Rule 13.2, second sentence, no technical relationship within the meaning of PCT rule 13 between the different inventions can be seen.

Consequently it appears that, claims 1, 2 and claims 3-5 (or claims 6-8, claims 9-22) do not satisfy requirement of unity of invention.

The feature common to both of claims 1, 2 and claims 25-46 is document conversion.

However, it is not necessary to exemplify and document conversion is common knowledge before this application.

Consequently the common feature is not a special technical feature within the meaning of PCT Rule 13.2, second sentence, since it makes no contribution over the prior art.

Therefore, there is no feature common to all the claims. Since there is no other common feature which can be considered as a special technical feature within the meaning of PCT Rule 13.2, second sentence, no technical relationship within the meaning of PCT rule 13 between the different inventions can be seen.

Consequently it appears that, claims 1, 2 and claims 25-46 do not satisfy requirement of unity of invention.

Furthermore, since claims 23, 24 are practically equivalent to a claim 1, the technical relationship within the meaning of the PCT rule 13 between these inventions can be seen.

フロントページの続き

(81)指定国 AP(BW,GH,GM,KE,LS,MW,MZ,NA,SD,SL,SZ,TZ,UG,ZM,ZW),EA(AM,AZ,BY,KG,KZ,MD,RU,TJ,TM),EP(AT,BE,BG,CH,CY,CZ,DE,DK,EE,ES,FI,FR,GB,GR,HU,IE,IS,IT,LT,LU,MC,NL,PL,PT,RO,SE,SI,SK,TR),OA(BF,BJ,CF,CG,CI,CM,GA,GN,GQ,GW,ML,MR,NE,SN,TD,TG),AE,AG,AL,AM,AT,AU,AZ,BA,BB,BG,BR,BW,BY,BZ,CA,CH,CN,CO,CR,CU,CZ,DE,DK,DM,DZ,EC,EE,EG,ES,FI,GB,GD,GE,GH,GM,HR,HU,ID,IL,IN,IS,JP,KE,KG,KM,KP,KR,KZ,LC,LK,LR,LS,LT,LU,LV,MA,MD,MG,MK,MN,MW,MX,MZ,NA,NI,NO,NZ,OM,PG,PH,PL,PT,RO,RU,SC,SD,SE,SG,SK,SL,SM,SY,TJ,TM,TN,TR,TT,TZ,UA,UG,US,UZ,VC,VN,YU,ZA,ZM,ZW

(72)発明者 藤巻 祐介
徳島県徳島市ブレインズパーク 株式会社ジャストシステム内
(72)発明者 檜山 正幸
東京都目黒区上目黒１－１４－３ ＳＴビル桜橋２０１

Ｆターム(参考) 5B009 QA06 SA13 TA11