2F－5

情報端末のための複合ドキュメント分散共有方式

羽根 秀宜, 河村 元夫, 横田 実
NEC C&C研究所

1. はじめに

インタネットに代表されるネットワーク技術の普及に伴い、WWWのようなネットワークワイドな情報サービスが盛んに利用されるようになってきた。しかしまだ容易に情報を分散して共有ができる枠組みは提供されていない。

そこで、我々はマルチメディアを含む複合ドキュメントを複数のユーザで分散共有できることを目指した情報端末の研究開発を進めている。

本稿ではこの情報端末における複合ドキュメントの分散共有の実現手法を提案する。まず目的とする複合ドキュメントとその分散共有形態について述べる。次にCILab[1]が中心に開発しているコンポーネントウェア技術OpenDoc[2]を用いた実現手法を述べる。更に、情報共有のためのオブジェクトモデルとして我々が提案しているShaped Object Model[3]を適用した実現手法を述べる。

2.複合ドキュメントと分散共有形態

既にOpenDocやOLE2[4]といったコンポーネントウェア技術によってマルチメディア情報を自由に扱える複合ドキュメントが実現されようとしている。しかしこれらの分散共有化はまだ実現されていない。

この複合ドキュメントに対して、我々が実現したい分散共有の要件を以下に示す。

1. 複数の端末からリモートにアクセスできること。
2. 複数の端末から例えばドキュメントの共同執筆や分散プレゼンテーションといった複合ドキュメントの同時共有ができること。
3. コメントの上書きなどのユーザ毎のカスタマイズや適切な排他制御といった柔軟な共有が行えること。

このための実現方式を次節で提案する。

3. 分散共有された複合ドキュメントの実現方式

3.1 Shaped Object Modelの概要

このモデルの基本的なアイデアはオブジェクトの外観とオブジェクトが本来実行する部分の分離であり、前者をShape、後者をBodyと呼ぶ。オブジェクトがリモートマシンからアクセスされると、そのマシンへShapeを転送しそのShapeとBodyとの間でメッセージ通信を行うことで分散オブジェクトを実現する。また一つのBodyを複数のShapeで共有することにより、オブジェクトの共有を実現する(図1)。

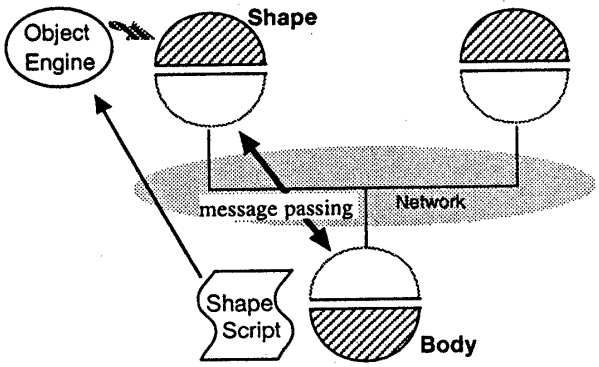

図1: Shaped Object Model

実際の実現では、ShapeはGUIを操作できるスクリプトで記述されており、リモートマシン上のスクリプトインタプリタで構成されるオブジェクトエンジンで評価/実行される。更にスクリプトを内容とするメッセージ通信によりShapeとBody間の機能連係を行う。

このShaped Object Modelには以下の利点がある。

- GUIに関わる細かい処理をShape側に分散することができる。
- オブジェクトのアクセス時にShapeをダウンロードするため、予めリモートマシン上にオブジェクトをインストールしておく必要がない。
- Shapeはスクリプトで記述されているので、プラットフォームに依存しない実現が可能である。

3.2 OpenDocの分散共有化[注1]

OpenDocでは、複合ドキュメントを構成するテキストや図形といった構成要素の表示/編集などの操作は、Partと呼ばれるコンポーネントプログラムにより行われることが特徴である。

OpenDocの構造を簡略化して書くと、図2のようになる。実行環境はSessionと呼ばれるオブジェクトが管理しており、各Partにはその描画領域に対応してFrameが与えられている。SessionからFrameの追加や削除、イベントの通知、描画要求がPartに送られ適切な処理を行う。ま

A method of Distributing and Sharing Compound Documents for Information Terminal
Hidetaka Hane, Motoo Kawamura, Minoru Yokota
C&C Research Laboratories, NEC Corporation

[注1] OpenDocではCORBA準拠の分散オブジェクト技術を用いた分散化が考慮されているとなっているが、まだその詳細は明確化されていない。

たドキュメントの実際のデータはDocument Storageに格納されており、Partの起動/終了時に読み込み/書き込みが行われる。

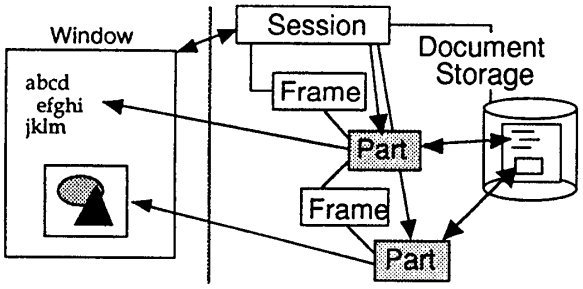

図2: OpenDocの構造

このような構造を分散共有化するために、次の点を考慮した。

1. 共有を行うために、少なくともDocument StorageおよびPartのGUI処理以外の部分はサーバ側に置くべきである。
2. Frameやその他のGUI関係のオブジェクトは、GUIの良好なレスポンスや端末毎のGUIコンテキストの確保のために端末側に置くべきである。
3. 1.2.を実現するために、Sessionをサーバ/端末それぞれに機能分割し、各オブジェクトの管理を行うようにする必要がある。

したがって、Partの機能分散をどのようにするかにより、次のような2つの方法が考えられる。

Case 1: Part全体をサーバ側に置く方法

Case 2: PartをModel-Viewのように分割し、Modelをサーバ側にViewを端末側に置く方法(図3)

Case 1ではレイアウト構造や描画プリミティブといった端末側のグラフィックレベルの情報を通信する必要があり、ネットワークの負荷が高いことが予想される。

Case 2はPartを分割するためCase 1に比べ変更部分が多くなるが、共有の制御のし易い枠組みや端末毎のViewのカスタマイズができる枠組みを提供できることになり、目標とするドキュメントの共有を実現できる。そこでCase 2を採用することにした。

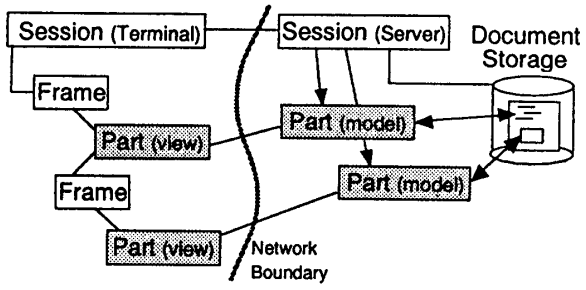

図3: Case 2におけるOpenDocの分散共有

3.3 Shaped Object Modelの適用

Partの分散共有を容易にするために、更にShaped Object Modelを適用することができる。PartのViewをShapeに、PartのModelをBodyに対応させることができる(図4)。

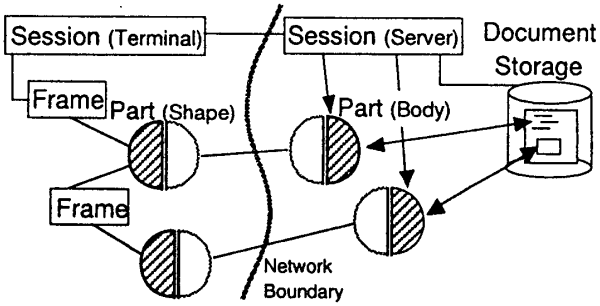

図4: Shaped Object Modelを適用したOpenDocの分散共有化

これにより、Shaped Object Modelの特徴であるオンデマンドなShapeのダウンロードという利点を得ることができる。つまり、テキストを編集するPartや図形を編集するPartといったようにいろいろな種類のPartにより構成される複合ドキュメントをアクセスする際に、そのPartのViewを端末上にダウンロードできる。したがって、端末上にそのPartのViewが存在しないためにドキュメントを正しく表示できないということが無くなる。

また、ここではPartの分割部分にのみShaped Object Modelを適用したがSessionの分割にもShaped Object Modelを適用できると考えられる。これにより複合ドキュメントのシステム全体をオンデマンドにダウンロードできることになり、システムのバージョンアップ等にも対応できる。

4 まとめ

複合ドキュメントの分散共有方式を実現するためにOpenDocをベースにShaped Object Modelを適用する分散共有方式を提案した。今後は、端末上のスクリプトインタプリタとOpenDocの関係など、Shaped Object Modelを適用する場合の詳細な問題点を整理し、実際に試作を行い評価を進めていく予定である。

最近注目されているJava[5]はインターネット経由でプログラムバイトコードをリモートマシン上に送信しバイトコードインタプリタ上で動作させる枠組みである。リモートマシン上にプログラムを転送し動作させる点ではShaped Object Modelとよく似たアイデアである。本稿で提案した複合ドキュメント分散共有方式のJavaを用いた試作も検討中である。

参考文献

[1] Component Integration Laboratories, http://www.cilabs.org
[2] Apple Computer Inc., "OpenDoc Programmer's Guide", 1995
[3] 横田, "Shaped Objectによる情報の分散共有", システムソフトウェアとオペレーティングシステム, 71-13, 1995
[4] Microsoft Corp., "INSIDE OLE2", アスキー出版, 1994
[5] Sun Microsystems Inc., "The Java Language: A White Paper", 1995