リアクティブ データフロー・モデルに基づく ドキュメントブラウザの設計と実現

5J−2

福田 晴元　高橋 直久
NTTソフトウェア研究所

1 はじめに

ソフトウェアの開発では，ソースコードをはじめ，様々な技術文書が作り出される．ソフトウェアの改造では、これら生産物（本稿では「ドキュメント」と呼ぶ）の中から必要なドキュメントを捜し出し、分析する作業が重要である。

このような分析作業を容易にするため，ハイパテキスト・システムがソフトウェア開発支援環境でも利用されている[1]．ドキュメントの開発時や分析時に，各ドキュメントに対して，関連するドキュメントへのリンクを付与しておけば，リンクを辿ることによりドキュメントを順に読み進められる．しかし，リンクの辿り方は，ユーザに任されているため，開発者や分析者の意図が十分に伝えられず，ユーザは試行錯誤を繰り返してしまうという問題が生じる．一方，従来の多くのオーサリングシステムと同様に，シナリオとして表示順にドキュメントを列挙しておく方法では，ドキュメントの数が多い場合には，分析過程の把握や変更が難しくなる．

本稿では，上記の問題を解決するため，ハイパテキストに対して、リアクティブ機能[2]を備えたデータフロー計算モデルに基づいてリンクの辿り方に関する意味付け法を与える，リアクティブ・データフローモデルと呼ぶドキュメント並行表示制御モデルを提案し，実現法を示す．また，このモデルに基づいて作成中のドキュメント・ブラウザ実験システムについて述べる．

2 リアクティブ・データフローモデル

リアクティブ・データフローモデルでは，ドキュメントの表示順序制御を行なうデータフロー型並列プログラムの実行過程としてドキュメントの分析過程をモデル化し，プログラムの再実行により分析過程を再現する機構を与える．このモデルでは，ドキュメントの表示過程を忠実に再現する機構とユーザとのインタラクションにより，適応的に再現法を変化させるリアクティブ機構とを統一的で簡明な制御モデルで実現することを目指す[3]．さらに，データフロー計算モデルにおける順序制御，非同期並列制御，手続き呼び出しなどの制御機構により，ドキュメントの分析過程が簡潔に表現できるようにすること，および，分析過程の表現の誤りの検出を容易にすることを目標とする．

2.1 データフロー型実行モデルによる並行表示制御

ドキュメントの分析過程をデータフローグラフで表現する．このグラフのノードは，ドキュメント・ノードと制御ノードからなり，アークはドキュメントの表示順序関係を表す．ドキュメント・ノードの「実行」とは，ノードに対応するドキュメントを表示し，編集作業を行なえるようにすることを意味し，実行終了後にトークンと呼ぶ制御信号を出力アークに送る．また，制御ノードは，分岐制御，手続き呼び出しなどの制御を行なうため，トークンの流れを制御する．

データフロー型計算では，トークンの授受により実行順序を制御し，全ての入力アークにトークンが送られたノードをトークンが揃ったノードといい，実行可能なノードとみなす．データフロー型計算の並列実行モデルである，データ駆動型および要求駆動型の実行制御法に基づき，データフローグラフを実行させると，それぞれ次のようにドキュメントの分析過程が再現される．

データ駆動型実行　トークンが揃ったノードを実行させる．この結果，ドキュメントが先頭のノードから順に並列表示される．

要求駆動型実行　指定されたノードを起点に先頭ノードまで，グラフの出力側から入力側にデマンドを伝播させる．次に，デマンドが伝播し，トークンが揃ったノードを実行させる．この結果，指定されたノードに到達可能なノードに対応するドキュメントだけが順に並列に表示される．

データフロー型実行モデルに基づきドキュメントの表示制御を実現することにより，次のような利点が得られる．

- 表示過程を構造的に分かり易く表示できる
- 複数のドキュメントが同期的に表示される．
- ドキュメントの表示過程が忠実に再現される．
- 要求駆動実行により，ユーザが指定したドキュメントの説明に必要なドキュメントだけが選択的に表示される．

2.2 リアクティブ機構による適応型表示制御

ユーザとのインタラクションにより再現法を変化させ，ユーザの置かれた局面に応じた表示を行うため，次のようなリアクティブ機構を実現する．

1. ユーザの選択したレベルに必要なドキュメントのみ表示するため，レベルに応じて表示過程を変更する機能
2. 補助的な表示過程を，ユーザの要求により，ある表示過程の再現中に実行する機能
3. ユーザの読むスピードに合わせて表示を行う機能

以下に，それぞれの機能の実現法を示す．

色付きアーク　データフロー計算モデルでは，色付きトークン方式により多重実行を可能としている[4]．この方式を利用して，ユーザのレベルに応じて表示過程を選択する機能を実現する．具体的には，ユーザのレベルを表すカラーをトークン，および，アークに付与する．カラーを持つアーク上では，同じカラーを持つトークンのみが伝幡する．これにより，特定のノードだけからなる部分的な表示過程が再現される．ユーザが自分のレベルのカラーをトークンに与えて実行を開始させた場合，ユーザはレベルに応じた表示過程を選択することとなる．

ユーザ依存アーク　ユーザの要求により，別の表示過程を再現するため，ユーザの要求を受けた時のみトークンを伝幡するユーザ依存アークを実現する．具体的には，アークにユーザ依存アークであることを示すマークを付ける．トークンがマークの付いたアーク上に到達した際に，グラフの実行機構はトークンを伝幡するかどうかをユーザに尋ねる．ユーザより要求のあった場合トークンを伝幡する．これにより，ユーザの要求があった時のみ別の表示過程を選択できる．

実行開始契機の制御　ユーザの読むスピードにあわせて表示を行うために，ノードの実行開始契機をユーザが与えるようにする．このために，システムは，トークンが揃い実行可能となっ

---

*Design and Implementation of a Document-Browser based on a Reactive Data Flow Model
Harumoto FUKUDA and Naohisa TAKAHASHI
NTT Software Laboratories

たノードがあることをユーザに知らせる．次に，ユーザより実行開始要求が来た後，実行可能なノードを実行する．これにより，ユーザは自分のペースで読み進める事ができる．

3 ソフトウェア設計ドキュメント・ブラウザ

3.1 構成法

図1にブラウザの構成を示す．本ブラウザで使用するドキュメントは，クラス，名前，型を持つオブジェクトとしてモデル化する．ドキュメントに，表示するツールごとにクラスを定義する．また，クラスごとにドキュメントの表示開始・終了メソッドを定義しておく．表示を行うためには，オブジェクトに対し表示のメッセージを送るだけでよく，表示用のツールを意識する必要がなくなる．また，一つのプログラムに関係のある全てのドキュメントに同じ名前を付ける．さらに，ドキュメントに対して，その用途ごとに型を割りつけ，プログラムが異なっても同じ用途で作成されたドキュメントが同じ型になるようにする．表示過程を作成するには，ノードに型を指定した抽象表示過程を作成する．次に，その抽象表示過程と名前の対を与えることにより，表示過程の名前を変えることで，グラフを再利用できる．

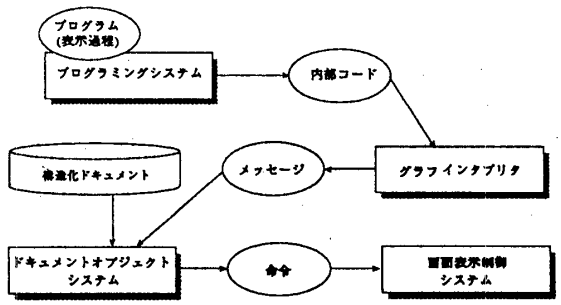

図 1: ブラウザ

以下に，図1に示した構成要素の各機能を示す．

プログラミングシステム　ユーザに表示過程を作成する環境を提供し，作成された表示過程を内部コードに変換する

グラフインタプリタ　リアクティブ データフローモデルの実行を行う．ドキュメントを表示するため，表示開始・終了，名前，型のメッセージをドキュメントオブジェクトシステムに送る．

ドキュメントオブジェクトシステム　名前と型で決まるドキュメントを，そのドキュメントの属するクラスのメソッドにより表示開始・終了を行うように，画面表示制御システムに命令を送る．

画面表示制御システム　ウィンドウの状態を管理する．送られた命令に従って，ドキュメントをその専用ツールにより開きウィンドウに表示する．

3.2 ドキュメント分析作業の進め方

本ブラウザを使用する際には，まず分析対象システムを開発した開発者，または，最初に分析作業を行った開発者が表示過程を作成する．このとき，分析に必要と考えるドキュメントを，機能，対象ごとにまとめ，後から分析する開発者が理解できる順に並べ，データフローグラフを作成する．次に，後続の開発者が本ブラウザを用いてデータフローグラフを実行させることにより，表示過程を再現し分析調査を進める．このとき必要に応じて，用意された表示過程に新たに表示するドキュメントを付け足してデータフローグラフを更新する．以上の作業を繰り返し，表示過程の作成と表示過程の再現によって分析作業を進める．

3.3 実現

Interleaf 社の DTP(Interleaf5) のカスタマイズ用の言語 Interleaf Lisp[5] を用いてブラウザを作成している．現在，インタプリタの機能の中で，データ駆動型実行により表示過程を忠実に再現する機構が稼働している．ドキュメントオブジェクトのメソッド，クラス管理と画面表示制御は Interleaf Lisp の機能を用いて実現している．図2に，簡単なデータフローグラフをデータ駆動型実行させた時に現れる画面例を示す．

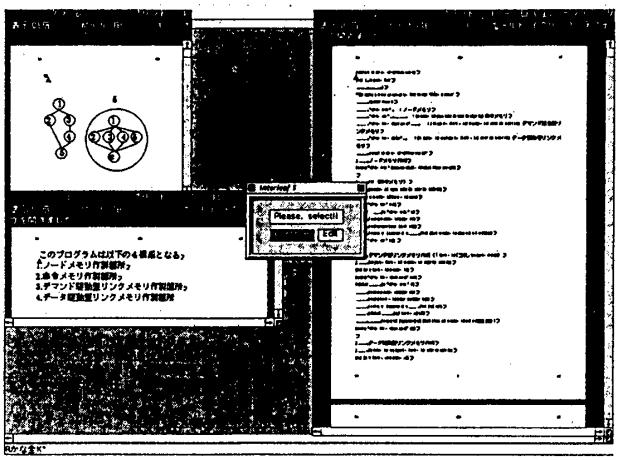

図 2: 実行例

4 おわりに

データフロー型実行機構とリアクティブ機構を融合したリアクティブデータフローモデルについて述べた．今後はドキュメントブラウザ実験システムの作成を進め，ブラウザの機能評価を行う．さらに，ドキュメントを構造化し，その構造を利用した表示法の考察，そして，分析作業の履歴データより表示過程を自動生成する手法の考察を行う．

謝辞

日頃御討論頂く伊藤正樹リーダはじめ，グループの皆様に深謝します．

参 考 文 献

1) 高田広章, ハイパテキストとそのプログラミング環境への応用, 情報処理, Apr 1989, Vol.30 No.4, pp406-413

2) D.Harel and A.Pnueli, *On the Dovelopment of Reactive System*, Logics and Models of Concurrent Systems, NATO ASI Series, Vol.F13 1985, pp477-498

3) 福田晴元 高橋直久, リアクティブデータフローモデル: ソフトウェア設計ドキュメントの表示過程の記述と再現, 情報処理学会 ソフトウェア工学研究会, Feb 1993

4) 雨宮真人, データフロー・アーキテクチャについて, コンピュータソフトウェア Vol.1 No.1 Apr1984 p.42-63

5) Paul M. English, Ethan S. Jacobson, and et al., *An Extensible, Object-Oriented System for Active Documents*, Proc. Int. Conf. on Electric Publishing, Document Manipulation & Typography Sep 1990, pp.263-276