リアクティブ・データフロー型ドキュメントブラウザの実現

2G－5

福田 晴元　高橋 直久
NTT ソフトウェア研究所

1 はじめに

我々は先に，ドキュメントの表示過程を並列プログラムとして記述し，そのプログラムの実行によりユーザのレベルに応じてドキュメントを選択的に表示する，リアクティブ・データフロー型ドキュメントブラウザを提案した．また，記述したプログラムにおける，構造及びレベル付けに関する妥当性を検証する手法を与えた．本稿では，ワークステーション上に実現したブラウザ(READY : Reactive Dataflow Document Browser System) の特徴について述べ，ドキュメント表示用プログラムの記述及び実行法について述べる．さらに，ドキュメント表示過程の再現機能とユーザ主導型の能動的な探索機能との融合法について述べる．

2 READY の特徴

2.1 READY の利点

READY は，ハイパテキストシステム[1)2)]におけるハイパリンクに対して，データフロー計算モデルに基づいてドキュメント間のリンクに対する意味付け法を与えているとみなせる．すなわち，ドキュメントの表示過程を，ドキュメントの表示順序制御，非同期並列表示制御，関数呼び出し等を行う構造的なデータフロー・プログラム (DFP) として記述可能にしている．DFP ではユーザの状況や基礎知識の違いをレベルの違いとして表現し，ドキュメントに対してレベル付けを行うことにより，各ユーザレベルに応じたドキュメント表示過程を一つの DFP により記述可能にしている．READY は以下の利点を持つ．

- DFP の実行時に動的にユーザレベルを設定することにより，レベルに応じた適切な表示過程を選択し実行できる．
- DFP の構造上の誤りやレベル付けの誤りにより生じる可到達性異常や再帰呼び出し異常等のデータフロー異常，及び，ネストしていない分岐構造等の構造誤りを検出する，プログラムの妥当性検証機能[3)]を持つ．

2.2 READY の構成

ドキュメント表示過程の再現系とプログラム検証系が同じ計算モデルに基づいているため，一つの実行系により実現可能である．これは，データフロー計算における発火規則と実行規則を一般化した一般データフロー計算[4)]を用いることにより実現した．図 1に示すように，一般データフローインタプリタに対してドキュメント表示を行うための発火規則と実行規則，及び，DFP を与えて実行させることにより，ドキュメント表示過程を再現する．また発火規則と実行規則を検証用の規則に入れ替えることによりプログラムの検証が可能となる．このように，READY では発火規則と実行規則を入れ替えるだけで，ドキュメント表示過程の再現系とプログラム検証系を同一のインタプリタで実現可能な構成となっている．

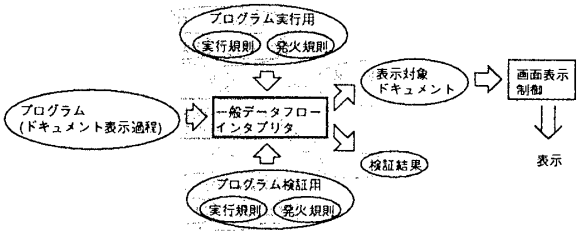

図 1: READY の構成

2.3 DFP の記述例と実行例

DFP は関数の集合として記述する．関数は関数名，入力ポートリスト，出力ポートリスト，関数本体からなる．関数本体はノードの集合からなる．ノードは出力ポートリスト，命令名，入力ポートリストからなる．実際の記述例を図 2に示す．ここでは，入力ポートを

```
DFP:
(program
    (function f1 (p1) (p7 p8)
        ((p2 doc "intro.doc" p1)
        ((p3 p4) distribute p2)
        (p5 doc "exp_exec.doc" p3)
        (p6 doc "exec.lsp" p4)
        (p7 doc "exp_verify.doc" p5 p6)
        (p8 doc "verify.lsp" p5 p6))))

レベル設定:
(set-level 'f1 'p1 3) (set-level 'f1 'p2 3) (set-level 'f1 '(p3 p4) 3)
(set-level 'f1 'p5 2) (set-level 'f1 'p6 3) (set-level 'f1 'p7 1)
(set-level 'f1 'p8 3)

実行:
(run-func 'f1 '(3) 1)
```

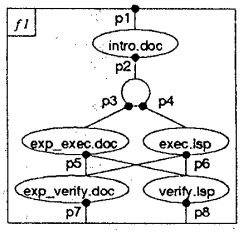

図 2: DFP 記述例とレベル設定及び実行法

p1，出力ポートを p7,p8 とする関数 *f1* が記述されている．DFP の 3行目に記述されたノードは，入力ポートはファイル名"intro. doc" と p1 ポート，命令名は doc，出力ポートは p2 ポートを指定している．リンクはポートのつながりにより表現する．実際には，4行目のノードの入力ポートは p2 であるため，3行目に記述されたノードと 4行目のノードはリンクでつながっていることとなる．各ノードの出力ポートに対し

Implementation of a Reactive Data Flow Document Browser System (READY)
Harumoto FUKUDA and Naohisa TAKAHASHI
NTT Software Laboratories

てレベルを与えることにより，ノードのレベルを設定する．例えば図のレベル設定例で (set-level 'f1 'p2 3) は，関数 *f1* のポート p2 を出力ポートとするノードをレベル 3 に設定することを意味する．実行を行う際には，最初に実行する関数名と初期値とユーザレベルの値をインタプリタに与えて実行を行う．

READY の主要部分は Common Lisp を用いて作成した．ウィンドウ表示制御は市販 DTP(Interleaf 5) の機能を利用している．上述のプログラム実行中のドキュメント表示例を図 3に示す．

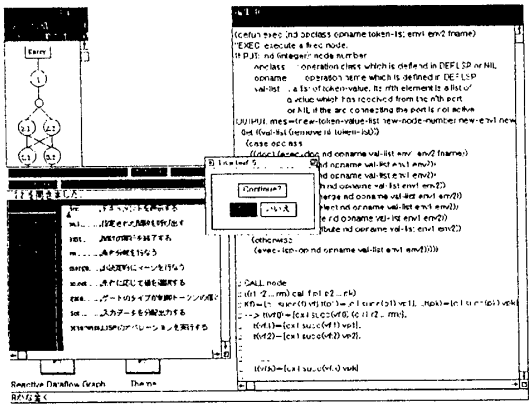

図 3: DFP 実行時のドキュメント表示例

3 ユーザ主導型探索機能との融合

READY では，必要なドキュメントの選択と関係付けは DFP の作成者が行い，ドキュメント表示制御は DFP を実行するインタプリタにより行われる．これによりユーザは必要なドキュメントを検索し，それらの相互関係を理解して必要な順に調べるといった作業をせずに，体系的な知識を得ることができる．

これに対して，ドキュメントの選択と関係付け，ドキュメント表示制御をユーザが行うことが必要となる場合がある．たとえば，注目しているドキュメント中の語句をキーワードとしてドキュメントを検索することにより，その語句の意味を調べたり，関連ドキュメントを探す場合である．

また，上記 2 つの形態の中間的なものとして，ドキュメントの選択と関係付けは既になされた状態で，ドキュメント表示制御をユーザが行う場合がある．これは，ドキュメントの選択と関係付けはリンクの形で実現され，そのリンクをユーザがたどることにより必要な知識を得る方法である．ここでは，DFP により表現されたドキュメント間のリンクも含むことができる．

そこで，上述の 3 種類のドキュメント表示手法を融合したブラウザを構築することにより，体系的な知識から非体系的な知識まで扱うことが可能となる．図 4 に，上述の 3 種類の検索手法の融合法を示す．図において，黒丸は現在注目しているドキュメントに対応するノードである．また，そのノードから見て次に表示可能なドキュメントに対応するノードを斜線で示す．図において class 0 はドキュメントをユーザが選択し表示制御もユーザが行う手法である．class 1 はドキュメントはリンクにより意味付けされているが表示制御はユーザが行う手法である．class 2 は DFP の実行によりドキュメントを表示する手法である．

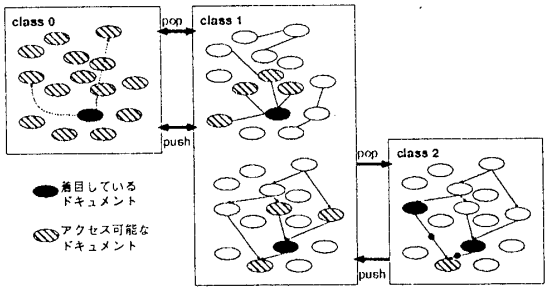

図 4: 3 種類のドキュメント表示手法の融合

以上の 3 種類の機能を融合するとき，正常に DFP の実行を終了させるためには，DFP の実行 (class 3 の状態) を中断してドキュメント検索 (class 0 または class 1) を行った後に，正確に元の DFP 実行環境へ戻る必要がある．また検索途中において，ある箇所に注目して検索を行った場合に，検索後にその箇所へ必ず戻って来られることが必要となる．

このために，環境を保存するためのスタックを用意し，ある class から他の class に移る際には，移動開始時点の環境をスタックに積む．検索終了後はスタックより環境を取り出し，順次元の class へ戻る (class 1 と class 0 の間のみ相互移動可能とする)．これにより，検索開始箇所へ必ず戻ることができる．スタックより全ての環境を取り出すと，プログラム実行環境へ戻ることができる．

4 おわりに

ワークステーション上に実現したリアクティブ・データフロー型ドキュメントブラウザ (READY) の特徴について述べ，さらに，ドキュメント表示用プログラムの記述例と実行例を示した．また，ドキュメント表示過程の再現機能とユーザ主導型の能動的な探索機能との融合法について述べた．今後はユーザ主導型探索機能を実現した READY に組み込み，実験的評価を行う．

最後に，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, Vol.30 No.4, pp406-413, Apr 1989.

2) P.DAVID STOTTS and RICHARD FURUTA University of Maryland, Petri-Net-Based Hypertext: Document Structure with Browsing Semantics, ACM Trans, Info, Sys,. Vol.7 No.1, pp3-29, Jan 1989.

3) 福田晴元 高橋直久, リアクティブ・データフロー型ドキュメントブラウザのためのドキュメントプログラミング, 情報処理学会 プログラミング - 言語・基礎・実践 14-3, Oct 1993.

4) 高橋直久 鈴木英明 直井邦彰 福田晴元, データフロー計算の一般化 - ソフトウェア・リエンジニアリングにおける 複雑な情報の構造化と理解を目指して -, 日本ソフトウェア科学会第 10 回大会, C9-2, Jun 1993.