

時間的に離れた協同作業環境を支援するための動作ビュー機構

藤田健治 上林弥彦

京都大学工学部

実時間型の協調作業支援システムにおいては協調作業を行なうユーザが同じ日時にシステムを利用しなければならないが、予定の都合や時差によってこのようなことが不可能である場合もしばしば起こり得る。このため他の利用者の作業中の様子を後で調べることにより協調作業を行なえるようにすることが重要になる。本論文ではこの機能を実現する動作ビュー機構を提案する。動作ビュー機構では過去の作業の記憶にビデオを用いず、利用者の操作と、それによる変化の起こった場合のみを記憶するという方式をとる。このようにすることにより特定の条件を満足する作業の一部を正規表現で表現し、効率よく検索することが可能になり、さらに大局的な検索を支援するために作業内の自動ワークフロー化の機能も実現している。

Action view mechanism to support asynchronous cooperative environment

Kenji FUJITA, Yahiko KAMBAYASHI

Faculty of Engineering, Kyoto University

In order to support user's cooperation under various environment, it is required to support cooperative work asynchronously. In this paper, a new function, called action view mechanism, is introduced for such a purpose. Basically, a user can replay the work done by other users when they are working together. Since recording is performed only when there are user's actions and related changes, drastic reduction of storage space is possible compared with conventional video recording. Furthermore, high-specification based on regular expressions to retrieve a required part. For global view of a user's work, generation of a workflow diagram is performed automatically.

1 はじめに

情報スーパーハイウェイの具体的な応用分野として地理的に分散した場所にいる利用者同士の協調作業支援が注目されている。この場合の1つの大きな問題として、個人個人のスケジュールの違いや時差によって必ずしも協調作業を行なう利用者同士が同じ時間にコンピュータを利用するとは限らない点があげられる。このため他の利用者の作業中の様子を後で調べることににより協調作業を行なえるようにすることも重要である。本論文ではビデオによらない過去の作業の記憶と再生の方式について述べる。この方法は、ビデオ方式よりもはるかにデータ量が少なく、また高度の検索機能や高度の機密保持機能が実現で出来るという点に特色がある。

会議などでは遅れて来た参加者や欠席者が会議の遅れを見返す時、画面出力のビデオを用いる方法が考えられる。この場合、データ量は膨大でデータ圧縮技術を用いてもCD1枚に60分程度しか収まらないため、現状ではビデオテープを用いる以外の方法は考えられない。ビデオテープを用いると、録画時間だけ再生時間がかかるので遅れてきた参加者はいつまでも会議に追いつくことが出来ない。このため、発言や操作のない部分のカット、倍速再生(音については高くならないような処理をする)、ダイジェストプレイ(サンヨー、音声はそのまま部分的にカットする)といった方法が知られているが、検索機能が十分とはいえない。

特に協調作業を行なう場合、相手の仕事のうち自分に関係するものを効率よく見つける必要がある。また目的にあった速度で再生・停止・早送り出来る機能とともに、全体の作業の流れを大局的に把握出来る機能も必要になる。

本稿で提案する動作ビューは過去の画面をビデオ画像として記憶するのではなく、利用者の操作と、それによる変化の起こった場合のみを記憶す

るという方式をとる。この方式の特色は次の通りである。

1. 記憶に必要なデータ量が大幅に減少する。
2. 画像の変化の理由が記録されるため、高水準の検索が可能である。ここでは正則表現を用いた検索言語を提案している。
3. 高度の機密保持が可能である。協同作業者の作業のうち関係のないものについては削除して示すことになる。
4. 実際の作業履歴より作業の関連を表すワークフローを作成する。このことから大局的な検索も可能である。

以下、第2章では動作ビュー機構の概略について、第3章では動作ビュー機能を利用できるアプリケーションプログラムに必要な機能について、第4章では動作ビューの検索を支援する検索言語の文法及び意味について、第5章では作業内容の概要の把握に用いられるワークフロー合成の方法について述べる。

2 動作ビュー機構の概略

動作ビュー機構を導入するシステムについて次のような仮定をおく。

- オブジェクト指向データベースを基礎とし、データの格納及びプログラムの実行はすべてデータベース側で行なう。
- ユーザはGUIを用いて操作をし、その操作はオブジェクト指向データベース内のオブジェクトへのメッセージに変換する。

動作ビュー機構ではユーザの操作によって直接データベース中のオブジェクトに送られたメッセージごとにそのメッセージの種類及びその結果起こったオブジェクトの変化や画面の動きを履歴に

取っておく。履歴から特定の場面の検索を行なって再生する画面ために検索言語を用意している。この検索言語では一連の操作の検索を指定するために正則表現を用いている。さらに、作業の概要を把握して検索に役立てるために、作業の履歴をワークフロー状にグラフ化して作業の流れを表示する機能を持たせる。動作ビュー機構を持ったシステムの全体像を図1に示す。

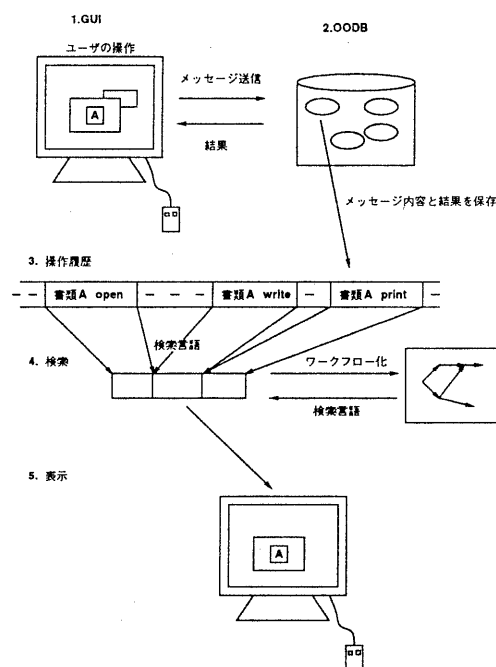


図 1: 動作ビュー機構を持ったシステムの全体像

3 アプリケーションモデル

本章では動作ビュー機構が利用できるアプリケーションプログラムに必要な機能について述べる。

3.1 過去のオブジェクト

前章で述べたように動作ビュー機能を導入するシステムはオブジェクト指向データベース上で構

築され、そのシステム内のアプリケーションの振舞いはデータベース内のオブジェクト群によってなされる。

過去の特定の時点の画面を再現は、履歴を元にその時点でのオブジェクト群を新たに生成して、それらに対して再生を行なうように動作ビュー機構がメッセージを送ることによって行なう。オブジェクトはオブジェクト ID を持っているが、再現用に作られたオブジェクトと現在のオブジェクトが同じ ID を持つことが有り得る。そのため、全てのオブジェクトにオブジェクト ID の他にビュー ID と呼ぶ属性をつける。現在のオブジェクトに対してはビュー ID を空値にしておく。後に説明する一つのビュー検索文の結果の再生に用いるオブジェクトには共通のビュー ID をつける。それらのオブジェクト ID は元のオブジェクトのオブジェクト ID と同じものをつける。

3.2 メソッド

オブジェクトが持つメソッドを次のような2種類に分類する。

外部メソッド ユーザからの操作によって直接メッセージが送られてもかまわないメソッドを外部メソッドと呼ぶ。

内部メソッド 間接的に他のメソッドの中から呼ばれることしか許されないメソッドを内部メソッドと呼ぶ。

ユーザの操作はあるオブジェクトに対する外部メソッドに対応するメッセージ送信に変換される。

3.3 操作履歴の形式

過去の操作を振り返ることが出来るようにするためには過去の操作の履歴を保存しておく必要がある。

ユーザの操作によって送られたメッセージによって起動された外部メソッドの中で、以下の項目を履歴の中に保存しておく。

1. メッセージを受けたオブジェクト (レシーバ) の ID
2. レシーバに対して送られたメッセージとその引数のオブジェクトの ID
3. このメッセージによるオブジェクトの属性値の変更点
4. この操作による画面の変化やマウスの動き
5. 操作が完了した時刻
6. 参照したオブジェクトの ID
7. 書き込んだオブジェクトの ID

計算途中のオブジェクトの属性値の細かい変化や一時的に作られるオブジェクトの情報は保存すると、記憶量が膨大になる。このためオブジェクトの属性値はメソッドの実行前と実行後の差分のみを保存しておく。また、間接的に呼ばれる他の外部メソッドや内部メソッドについてはこのような履歴は残さない。

全ての外部メソッドに履歴を作成するための命令コードを明示的に挿入する。挿入される命令コードの概要を以下に示す。

メソッドの先頭部

```
if (ユーザから直接呼ばれたメソッド)
```

```
begin
```

1. 2. の項目を履歴に書く。

このメソッドでの処理前のオブジェクトの状態を保存する (3 の項目)。

このメソッドを起動するまでのマウスやキーボードの操作を書き込む (4 の項目)。

```
end
```

メソッドの終了部

```
if (ユーザから直接呼ばれたメソッド)
```

```
begin
```

このメソッドでの処理後のオブジェクトの状態を保存する (3 の項目)。

このメソッドの処理の結果起こった画面の変化を保存する (4 の項目)。

5. に現在時間を書き込む。

6. 7. の項目を履歴に書く。

```
end
```

履歴の 4. の項目には 1. から 3. では操作の再生に不足する情報を全て書き込んでおく。これは後に外部メソッドに一对一に対応する再生を行なうメソッドでのみ用いられる。この項目に詳しく記録すればするほど再生した画像がもとの画像に近くなる。例えば現在作成中のプロトタイプシステムでは、マウスの座標をクリックした座標のみ保存している。その結果、再生する時のマウスの移動経路は元の経路と異なりシステムが補間したものになる。

データベースに時間の概念を入れる試みとして temporal database があり、その SQL 言語として TSQ2 が提唱されている [4]。TSQ2 を拡張利用することによって検索機能の一般化が可能である。

3.4 再生のメソッド

過去の画面の再生のために全てのオブジェクトに次のような内部メソッドを用意する。

1. 属性値を元にその時の画面を再生するメソッド
2. 全ての外部メソッドに一对一に対応する、履歴を元に対応するユーザの操作を再生するメソッド

4 検索言語

記録された画面の検索を支援する機能としての二種類を用意する。

- 検索言語を用いて時間的、空間的範囲を指定させる。
- 作業内容からワークフローを合成して表示し、その上で検索場所を指定させる。

本章では協同作業者の操作の系列の内の動作ビューを通じて見たい範囲を指定する検索言語について述べる。ワークフローの合成法については5章で述べる。

操作はメッセージに変換されるので単一の操作はメッセージとメッセージが送られたオブジェクトを指定することで指定できる。一続きのメッセージ列の検索のためにメッセージ列を単一の操作指定の正則表現で指定出来るようにする。

4.1 検索言語の文法

図2に文法の定義を示す。動作ビューの検索言語では次のような項目を指定する。

1. <操作指定> によって単一のメッセージを指定する。次のような項目からなる。
 - (a) メッセージを受けるオブジェクト (レシーバ) のクラスの指定

(b) メッセージの名前

(c) レシーバ、メッセージの引数の属性値などの条件

2. 操作名の正則表現によって動作ビューで検索するメッセージ列を指定する。
3. 時間の指定。特定の時間の範囲から検索することを指定する。

```
<ビュー検索文> := <操作指定> * <検索式> <時間>

<操作指定>      := define operation <操作名>
                  class <レシーバのクラス名>
                  message <メッセージ名と仮引数名>
                  condition <条件式>

<検索式>         := search expression
                  <操作名> による正則表現

<時間>           := time interval
                  from <開始時刻> to <終了時刻>
```

図 2: 動作ビュー検索言語の文法

4.2 検索文の意味

操作名 A の操作指定があつて検索式が A である場合は履歴から操作指定の条件に合うメッセージ送信全てを検索結果とする。例として操作履歴が $a_1 b c a_2 d e f a_3$ (a_i は A の条件に合う操作であるとする) であった場合検索結果は $a_1 a_2 a_3$ となる。

操作名 A, B の操作指定があつて検索式が AB である場合は操作履歴のなかで操作指定 A の条件に合う操作 a があつてその後に a の操作の対象となるオブジェクトと関係がない操作が続いた後に操作指定 B の条件に合う操作 b が続くような操作のペア ab 全てが検索結果となる。例えば $a_1 b_1 c b_2 d e f a_2 a_3 g h i b_3$ の場合の検索結果は $a_1 b_1 a_3 b_3$ となる。

操作名 A, B の操作指定があつて検索式が $A|B$ である場合は操作履歴のなかで操作指定 A の条件に合う操作 a があるかまたは操作履歴のなかで操作指定 B の条件に合う操作 b がある時、条件に合う操作全てが検索結果となる。例えば履歴が $a_1c_1db_1b_2efa_2a_3ghib_3$ で検索式が $A(B|C)$ であるとき、結果は $a_1c_1a_3b_3$ となる。

A^* は次節で述べる変数のスコープを除いて $\sum_{i=0}^{\infty} A^i$ と同等である。

4.3 変数

条件式では仮引数名の他に、次のような定義済み変数を用いることが出来る。

<操作名> メッセージのレシーバ (メソッド送信前)
<操作名>.new メッセージのレシーバ (メソッド送信後)

他の変数名は定義せずに自由に使うことができる。操作指定による正則表現における変数のスコープは以下のようにする。

- AB

正則表現 A で変数 x が初めて用いられており値が代入されていたら、 B で用いられている x は A 中の x と同一の値を表す。

- A^*

正則表現 A の内部で変数 x が初めて用いられているとする。それぞれの A にマッチする操作指定列の条件式の x の値は同一の値を表す。繰り返しの間では変数は共有しない。

- $A|B$

正則表現 A と正則表現 B で初めて用いられる変数 x があつた場合、 A の x と B の x には関連はない。

図 3 に動作ビューの検索言語例を示す。

4.4 検索結果の表示

検索結果の表示は以下のような手順で行なわれる。

```
define operation A
  class Text
    message copy: aPosition1
    condition (A name = 'document1')
```

```
define operation B
  class Text
    message paste: aPosition2
    condition (B name = 'document2')
```

```
define operation C
  class Printer
    message print: B
    condition
      search expression (AB)*C
```

```
time interval
  from 'pm6:00 yesterday' to now
```

図 3: 動作ビューの検索言語の例 (document1 の一部を document2 にコピーしてプリンタに出力した様子を探す)

1. $M \leftarrow$ 操作履歴から本章で定義された検索言語によって得られた表示すべきメッセージの集合
2. $O \leftarrow M$ の要素のレシーバまたは引数に現れるオブジェクトのオブジェクト ID の集合
3. $m \leftarrow M$ の最初の要素
4. 全てのオブジェクトについて同じオブジェクト ID を持ち m の前の時点での属性値を持つ新たなオブジェクトを作る。それらのビュー ID は同じ値にする。

5. 全ての O の要素に再生のメソッドを適用し、この時点での画面を作り出す。
6. m が示すメソッドに対応する再生のメソッドを適用し、このメソッドが起動された様子を動画で表示する。
7. $m \leftarrow M$ の次の要素。なければ終了。
8. 5. へ行く。

過去の操作を見る場合、何もしていない場面や動作ビュー検索言語で表示するように指定されていない操作を行なっている場面は省略して表示することも出来る。省略している間のマウスの動きは前の操作の終了時の座標から次の操作の開始時の座標まで直線で補間することで自然な画面を提供する。

4.5 検索言語入力インターフェース

これまでに述べた検索言語はエンドユーザ向けではない。メニューなどを多用して検索言語を入力する使いやすいインターフェースの開発が今後必要になると考えられる。

5 作業内容からのワークフロー合成

行なわれた作業の流れをワークフローとして表示することで作業内容の概略を把握や検索の役に立つことが考えられる。本章では作業の履歴を元にワークフローを合成する手法を述べる。

5.1 1つのメッセージのワークフロー化

あるメッセージ送信によってメソッド M が起動され M の中でオブジェクト $I_1, I_2, \dots, I_n$ が読み込まれられ、オブジェクト $O_1, O_2, \dots, O_m$ に書き出されたとする。 $I_{1 \sim n}$ と $O_{1 \sim m}$ に共通のオブジェクトがあってもよいものとする。

このメッセージ送信を図4のように表すことにする。

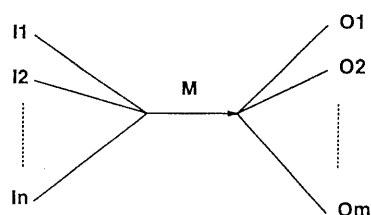


図 4: 1つのメッセージのワークフロー化

5.2 複数のメッセージのワークフロー化

複数のメッセージをグラフ化するときには、それぞれのメッセージを前節の方法によってグラフに変換しそれらのグラフをメッセージが送信された時間順に合成していくことによっておこなう(図5)。

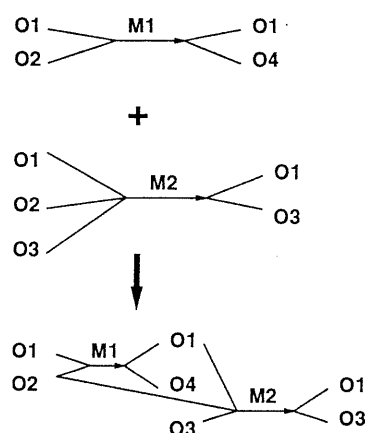


図 5: 複数のメッセージのワークフロー化の手順

合成の手順は次のようにする。

1. 新たに追加するグラフの入力ノードそれぞれについてループ
2. その入力ノードがすでにあれば最も新しいものにつなぐ
3. 入力ノードがなければ新たに作る

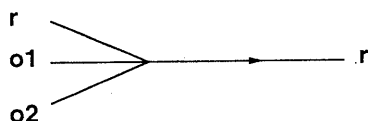
5.3 メソッドでの入出力オブジェクトの決定法

次のようなSmalltalk式を考える。

```
r par1:o1 par2:o2
```

これはオブジェクトrに対して二つのオブジェクトo1,o2を引数としてpar1:par2:というメッセージを送ることを意味する。

副作用がなければこのメソッドに対する入力オブジェクトはr, o1, o2、出力はrと考えることができる。



しかし一般にはメソッド中でメッセージを受けたオブジェクト、パラメータ以外のオブジェクトを参照したり、メッセージを受けたオブジェクト以外に書き込むことが有り得る。参照したり書き込んだオブジェクトを定める方法として次の2通りを考えている。

1. メソッドごとに参照、書き込みを行なうオブジェクトをあらかじめ決めておく
2. 実行時に参照、書き込みを行なったオブジェクトを記録する。

5.4 ワークフローの一部の取りだし

4章で定義した動作ビューの検索言語を利用することによって特定のメッセージの集合を得ることができる。この集合に含まれるメッセージのみをワークフローとして可視化することでユーザがワークフローの一部の取りだすことができる。

5.5 ワークフローの簡略化

これまでに述べた手法で合成したワークフローは一般に大変数が多くなり見て理解をする事は困難になる。そこで、アプリケーションの起動や

終了などに注目してワークフローを階層化し下位のワークフローを簡略化して表示することによって、より抽象的なワークフローを得ることが出来ると考えられる。これによってユーザが作業の全容をより把握出来ると考えられる。

6 おわりに

本稿では協同作業環境において作業内容の画面を記録、検索、再生する機能を持つ動作ビュー機構について述べた。動作ビュー機能は当研究室で開発中のオフィスにおける協同作業支援システム[3]の一機能としてインプリメント中である。履歴の記憶量、検索の時間、および使いやすさの評価は今後の課題である。

参考文献

- [1] 向山 博: 協調作業支援技術の研究開発動向、情報処理学会ソフトウェア工学研究会 1993.7.8, p85-92
- [2] 石井裕: リアルタイムグループウェアのデザイン、情報処理学会誌 Aug. 1993, p1017-p1027
- [3] Hideyuki Takada and Yahiko Kambayashi: An Object-Oriented Office Space Description Model and an Office View Management Mechanism for Distributed Office Environment, FODO'93 p362-377, Springer-Verlag
- [4] Richard T. Snodgrass et.al. : TSQ L2 Language Specification, p65-p96 SIGMOD RECORD, Vol23, No. 1, March 1994