Java アプレットのためのアニメーションヘルプ作成システム

三浦元喜　　田中二郎
筑波大学 理工学研究科　　筑波大学 電子・情報工学系
miuramo@softlab.is.tsukuba.ac.jp　　jiro@softlab.is.tsukuba.ac.jp
305-8573 茨城県つくば市天王台 1-1-1 筑波大学電子・情報工学系 田中研究室
Tel: 0298-53-5165　　Fax: 0298-53-5206

1　はじめに

ヘルプ機能は，アプリケーションの操作方法が分からないユーザのために必要である．従来は文章によるヘルプが一般的であった．コマンド入力を主体とするアプリケーションでは文章によるヘルプでも問題はない．しかし，GUI (Graphical User Interface) を用いたアプリケーションにおける操作方法は，文章による説明ではユーザの負荷が大きい．例えば，ユーザは「『ファイル』メニューの『開く』を選択」のような文を読み，アプリケーションの画面からそれらの「操作対象」を探して操作する必要があった．

文章の代わりに，アニメーションを用いてヘルプを提示する方法が提案されている [1] が，実装のための手間がかかるため，いままで実際のシステムにはあまり適用される例が少ない．

我々は，アニメーションヘルプを実現するための手間を減らすため，アプリケーションとヘルプ機能を切り離し，なおかつ簡単に統合できるための枠組みを，Java アプレットについて実装した [2]．Java アプレットは，Web ブラウザを用いれば誰でも実行できるが，その Web ページを初めて訪れる人にとってはアニメーションヘルプが用意されていればアプレットの概要を容易に把握できる．アニメーションヘルプは特に直接操作を多用したシステムの操作説明に有効であると考えられる．

このシステムを使うと，(1) 開発者は，ヘルプの対象となるアプレット (対象アプレット) における典型的な操作を行い，それを記録する．(2) ユーザは，開発者によって記録された操作を読み込み，アニメーションによって見ることで，具体的な対象アプレットの操作方法や概要が理解できる．我々はこれらの技術を総称して特に Rapid Help 技術と呼んでいる．

2　設計方針

我々は，開発者が作成したどのようなアプレットにも対応するため，ヘルプ記録 / 再生システムの部分を完全に対象アプレットから分離する方針を取った．ヘルプ記録 / 再生システムを分離すれば，対象アプレットのソースコード，バイトコードを変更しなくて済む．また，どの環境でもヘルプ再生ができるように，Java VM (Virtual Machine) には手を加えない．なおかつ，開発者が簡単にヘルプ記録 / 再生システムを組み込めるような仕組みが望ましい．

我々は，以上の点を考慮して，ヘルプ機能をアプレットビューアの機能を持つアプレット (ヘルプアプレット) として実現した．開発者は，パラメータとして対象アプレットのクラス名を指定するだけでアニメーションヘルプ機能を統合できる．

操作の記録　通常，対象アプレットで発生したイベントは対象アプレット内で処理されてしまう．ヘルプアプレットが対象アプレットで発生したイベントを取得するのに，専用のイベントリスナを用意する．このイベントリスナを部品に登録することで，ヘルプアプレットは対象アプレットのイベントを取得できる．対象アプレット初期化後，すべての部品に登録する．その後新たに部品が追加されたら，その部品について動的にイベントリスナを登録する．

操作の再現　記録されたイベントは，イベントが発生した部品に送り返すことで再現できる．イベントは本来発生した部品への参照を持っているが，この参照は一時的なもので再現時には無効となるため，我々はそのイベントがどの部品から発生したものかを特定する必要がある．部品を特定するために，我々はイベントを発生した部品の木構造における位置を示すパス情報をイベントが通知されたときに記録しておく．そのパス情報を用いて，部品の木構造を探索することで再現時に部品を特定することができる．

3　操作の意味付け

記録した操作はただ再現するより，**図 1** のように現在どの操作を行っているかを提示しながら見せた方がユーザにとって分かりやすい．これらの情報は開発者が入力してもよいが，新たにイベント列を記録する毎に入力するのは面倒である．

我々は，意味を持つイベント列を「コマンド」とし，そのコマンドにラベルとして意味を表す文字列を付け，それを提示することにした．しかし，任意のイベント列が対象アプレットでどのような意味を持つのかは，完全に分離されたヘルプアプレットには判別できない．対象アプレットがヘルプアプレットにイベント列の意味を通知するには，対象アプレットがヘルプアプレットへの参照を持つ必要があり，対象アプレットを変更せざるを得ない．

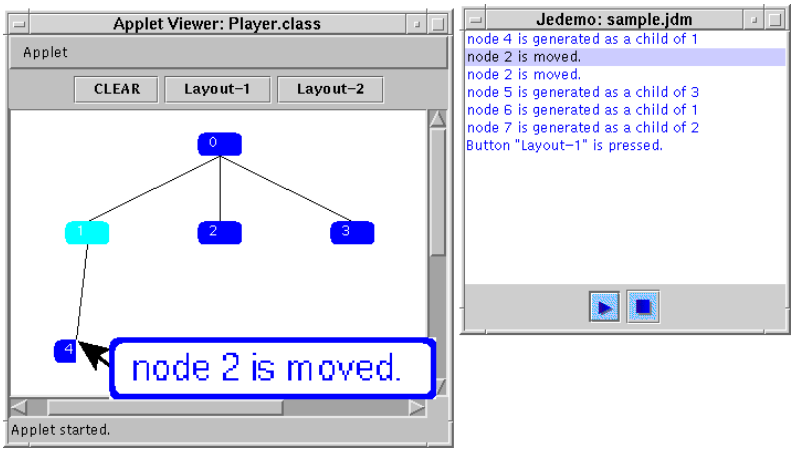

図 1: Player アプレットで操作を再現している画面

図 2: セパレータとコマンド候補

そこで，ヘルプアプレットが意味を持つイベント列を判別するために，我々はコマンドルールという生成規則を与える．コマンドルールには，イベント列を判定するためのイベントパターンと，ラベルを生成する規則 (ラベル付けルール) を持つ．1 つのコマンドルールは，1 種類のコマンドを判定し，生成する．

コマンドを生成する仕組みは以下のようになる．まず，対象アプレットから送られてくるイベント列をセパレータ (イベント列を区切るための特別なコマンドルール) によって分割する．イベント列には意味のある部分と意味のない部分が交互に現れる **(図 2 参照)** ので，そのうち意味のある部分 (コマンド候補) だけを抽出する．そのイベント列と，コマンドルールのイベントパターンとの照合を 1 つずつ行っていく．照合に成功した場合には，ラベル付けルールに基づいてラベルが付けられる．ラベル付けルールには，固定文字列だけでなく，「押した部品を特定する文字列」など動的な情報を挿入するため，特定のオブジェクトのメソッドを呼びだすことができる．例えば，

```
node @press.getNodeID() is moved.
```

と記述することでマウスが押されたノードの ID 番号をラベルに挿入でき，**図 1** のようにユーザに提示できる．

4 実装システムと手順

コマンドルールの生成は，実際の対象アプレットで発生したイベントを利用して簡単に行える．まず，開発者は，Recorder で対象アプレットの一通りの機能を操作し記録しておく．次に，Analyzer で保存したイベントを読みこみ，各イベントをアイコンで表示する．開発者がイベント範囲指定でセパレータを指定すると，コマンド候補とセパレータが交互に現れる．さらに，同じコマンドが同じコマンド候補になるようにコマンド候補を編集し，コマンドルールのイベントパターンを生成する．その後，CommandRuleEditor によって，それぞれのコマンドルールに対応するラベル付けルールを記述する．その後，Generator がコマンドルールを読み込み，操作からラベル付きのコマンドを生成する．

ユーザは，Player によってコマンドを見る．コマンドのラベルが一覧表示され，概要を把握できる．また，擬似マウスカーソルを動かしたりポップアップウィンドウによるメッセージを表示することによって，**図 1** のように誰かが対象アプレットを操作しているかのような感覚を与える．

5 まとめ

アプレットのためのアニメーションヘルプ作成環境を設計，実装した．ヘルプシステム部分を対象アプレットと完全に分離したことで，開発者にとってアプレットの効果的な操作説明を短時間で簡単に作成できる．またユーザにとっても直感的にアプレットの概要を把握できる．Rapid Help 技術は頻繁にバージョンアップされるアプレットには特に有効である．

参考文献

[1] Piyawadee Sukaviriya and James D. Foley: Coupling a UI framework with automatic generation of context-sensitive animated help, *Proceedings UIST '90*, pp152–166, 1990.

[2] Motoki Miura and Jiro Tanaka: A Framework for Event-driven Demonstration based on the Java Toolkit, *Asia Pacific Computer Human Interaction (APCHI-98)*, pp 331–336, July 1998.