GUI対話制御の構築を支援するObjectchartsエディタの実装

7H−5

佐藤 武秀, 増田 英孝, 笠原 宏
東京電機大学 工学部

1 はじめに

GUI (Graphical User Interface) を構築する効率を上げるためにUIMS (User Interface Management System) という考え方が提案されている。このUIMSの中でシーハイムモデルを基に考えられているものは、プレゼンテーション部、対話制御部、アプリケーションインタフェース (API) 部の3要素から成っている [1] [2]。

本研究は、この対話制御部の構築を、グラフィカルに支援することが可能なエディタを用いて、容易に実現するものである。具体的には、対話を定義するための記述法として既に提案されているStatecharts、Objectcharts [3] を用いたモデルについて検討を行ない、その場合に有効となるObjectchartsエディタを実装するものである。

本稿は、本Objectchartsエディタにあるレイアウト機能、シミュレーション機能を用いることによって、対話制御において対話の流れを決めるStatecharts、Objectchartsが簡潔に描け、さらに対話の流れをシミュレートすることも可能であることを示す。

2 対話定義の構築

対話制御の構築法は既にさまざまなものが発表されているが、その中でも対話定義において、対話の流れとの整合性の良さから状態遷移モデルがよく用いられる。しかし対話数が増大したり並列に処理する必要が生じると、状態遷移図は非常に複雑となり実用から遠ざかる。そこで状態遷移図に階層化、構造化、モジュール化機能を加え、さらに並列処理の記述法も加えたStatechartsをそれの代わりに使うことが考えられた。これにより、簡潔にかつ効率良く対話定義が構築出来るようになった [2]。

またObjectchartsは、Statechartsが状態遷移のみを記述するのに対し、状態遷移が記述出来るのに加え、状態の属性を報告する機能も持っている。

従ってStatecharts、Objectchartsを利用して対話定義を構築することが、状態遷移図から引き継いだ対話の流れとの整合性の良さと、階層化等の各機能を持ったことによる効率性の点から考えて非常に適していることが理解される。

3 Objectcharts エディタ

3.1 Statecharts、Objectcharts を用いての対話定義の構築時の問題点

実際にStatecharts、Objectchartsを用いて対話定義を構築する場合以下の問題が生じる。

1. Statecharts、Objectchartsは階層化機能、並列処理機能を持つため、紙面に描いていてはその機能を充分に活かすことが出来ない。

2. Statecharts、Objectchartsを用いても対話を追加、削除、変更するたびにチャート上の状態とイベントの配置を再検討する必要が生じる。

さらに対話定義を構築する場合、その構築の段階においてシミュレーションを行なうことが有効となる。

従ってStatecharts、Objectchartsがディスプレイ上で手軽に描け、シミュレーションも可能であるエディタ、シミュレータが望まれる。そこで、以下に述べる機能を持ったObjectchartsエディタを実装した。

3.2 本Objectcharts エディタの特徴

本Objectchartsエディタの特徴を実際の構築手順に沿って説明していく。

まず、

1. 状態の生成、状態名の決定、状態の配置

を行なう。これは、エディタ上の任意の位置でポップアップメニューから選ぶことで実行出来る。任意の位置で状態生成が可能であるため、ある状態の上にさらに状態を生成することが可能となる。この状態の親子関係を内部モデルに保存することにより、階層化が実現出来る。次に、

2. イベントの生成、イベント名の決定、イベントの設定

を行なう。ここでイベントを設定するということは、そのイベントが起こる前後の状態を決定するということになるが、この時にこのイベントとその前後の状態との関係が内部モデルに保存される。この内部モデルは状態やイベント

An Objectchart Editor for assisting implementation of GUI dialogue control
Takehide SATO, Hidetaka MASUDA, Hiroshi KASAHARA
Tokyo Denki University

に対する変更が起こった時と、シミュレーション時に参照され Statecharts、Objectcharts が常に整合性を持った動作をするように管理するものである。さらに、

3. デフォルト状態の設定

も行なう。これは、抽象状態と呼ばれる各状態集合間におけるデフォルトの状態を決めることである。

以上の 1〜3 を任意の順序で繰り返すことにより対話定義を構築していくわけであるが、その途中で実際の対話の流れを確認する必要が生じる。そこで、

4. シミュレーション

を行なう。シミュレート方法はいくつも考えられるが、本 Objectcharts エディタでは 2 方法用意した。1 つは 1 ステップずつ状態遷移を確認しながらシミュレートするものであり、ポップアップメニューより希望するイベント名を 1 つ選択することで実行出来るものである。そしてもう 1 つは、1 つ以上のイベントをポップアップダイアログに連続して入力してから実行するものであり、1 度で対話の流れを確認出来るものである。

さらに 1〜4 の機能の他に状態名、イベント名の変更、状態の配置される位置の移動、リサイズ、削除も結合しているイベントの内部モデルを参照し整合性を持って行なうことが出来る。また、並列処理の記述も容易に実現出来る。

3.3 適用例

本 Objectcharts エディタを用いて対話定義を構築したので以下に示す。図 1 は、本 Objectcharts エディタにおけるイベント設定の対話モデルの Statecharts である。ここではシミュレーションを実行し、set event というイベントを与えたので現在の状態は WaitForInputOfName である。現在の状態は太線で描くこととした。

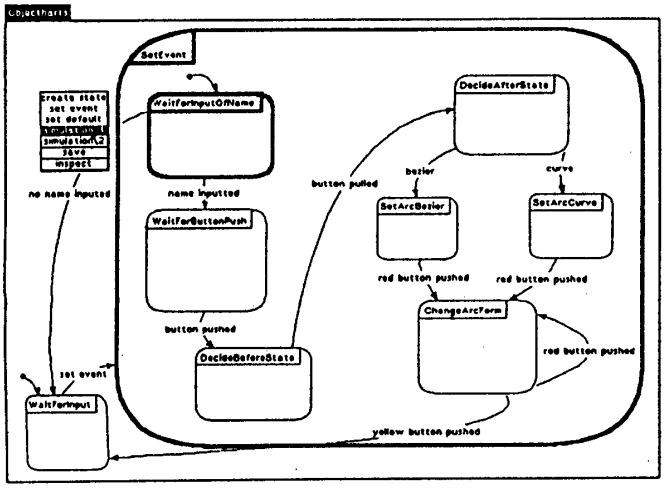

図 1: イベント設定の対話の流れ

図 2 は図 1 で構築された対話の流れを変更したものである。ここでは、イベント名の確認の対話を付け加えた。

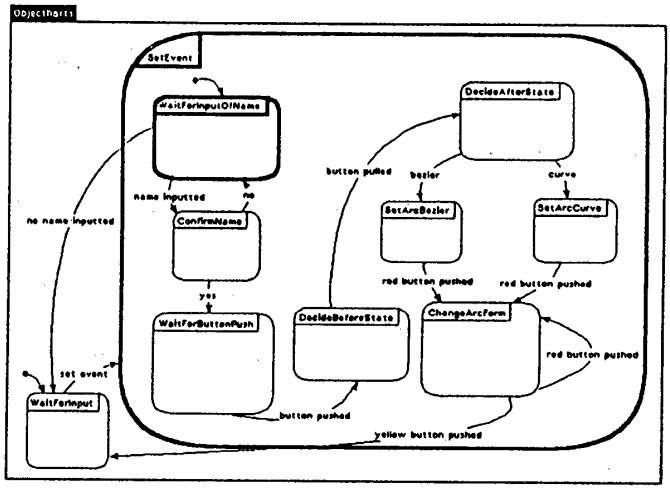

図 2: 変更後の対話の流れ

4 おわりに

今回は、Objectcharts エディタを実装し、それの持つ機能により対話定義の構築を支援していく方法について述べた。これにより対話定義が構築出来、シミュレートすることが可能となった。

今後は、対話定義がよりスムーズに構築出来るよう支援を強化し、最終的にはシーハイムモデルのプレゼンテーション部と API 部の関数を用意すれば、AP が要求する GUI の対話制御を容易に構築出来るエディタを実現するつもりである。

参考文献

[1] 萩谷. グラフィカルなユーザインタフェースとその開発環境について, bit, Vol.21, No.6, May 1989

[2] 宮崎. ユーザインタフェース管理システムと対話制御, 情報処理, Vol.33, No.11, Nov. 1992

[3] D.Coleman, F.Hayes, and S.Bear. *Introducing Objectcharts or How to Use Statecharts in Object-Oriented Design*, IEEE Trans. on Software Eng., Vol.18, No.1, Jan. 1992