6G−9

ペトリネットを用いたグラフィカルユーザインタフェースのふるまいの記述

上野 浩一郎，　田村 直樹，　中島 毅，　上原 憲二

三菱電機(株)　情報電子研究所

1　はじめに

アプリケーションに一層の使い易さを提供するために、グラフィカルユーザインタフェース(GUI)の需要が高まっている。このようなGUIの開発のため、GUIの部品を画面上で配置して、画面設計を行なうツールが多く提案されている[1]。しかし、これらのツールでは、ユーザの操作に対するGUIの動的な反応を捉えることが難しい。一方、GUIの動的な反応の記述に、表示画面の変化を状態遷移図で記述する方法がある。しかし、状態遷移図を用いて並列的なふるまいをするGUIの反応を記述することは困難である[2]。本報告では、ユーザの操作に対するGUIの動的な反応であるふるまいを、並列事象のモデル化に適しているペトリネットを用いて記述する方法を検討し、その記述実験を行なう。

2　GUIのふるまいの記述

2.1　記述の必要性

GUIの設計者は、部品を配置するツールを用いて、見ため（ユーザにGUIが画面上でどのように見えるか）を設計することができる。しかし、一般にGUIは、ユーザの様々な操作に適切に反応しなければならないので、見ための設計だけでなく、GUIのふるまい（ユーザの操作に対する反応）を設計することも必要である。配置ツールではこれに対する支援がない。

GUI部品の数とそれらに対する可能な操作の数が多くなると、システムの適切な反応を設計することは難しくなる。この難しさを軽減するためには、ふるまいを記述することが必要である。GUIのふるまいを視覚的で形式的な記述言語によって記述することによって、理解容易で誤りの少ない設計が可能になる。

2.2　記述上の問題点

一つのシステムのGUIは、同時に操作可能な複数の部品から構成される。各部品のふるまいは、部品に対するユーザの操作と部品のもつ状態のみによって決定されるのではなく、関連する他の部品の状態をも含めて決定される。

従来、ふるまいの記述には状態遷移図が用いられることが多い。GUIのふるまいを状態遷移図で記述する場合、通常は、GUIの全表示画面をステート、ユーザの操作をイベントとして捉える。ユーザの操作(イベント)と全表示画面(現ステート)から、次の全表示画面（次ステート）が決定する。部品の組合せからなるGUIのふるまいを状態遷移図で記述する場合、（本来部品間の相互依存性は小さいにも関わらず）各部品の状態のすべての組合せの数だけ状態を考慮しなければならない。そのため、考慮すべき状態数が非常に大きくなり、大きなシステムを記述することは事実上不可能となる。

一つのシステムのGUIのふるまいをモデル化する場合、まず、部品に対するユーザの操作と各部品の見ための変化を自然に記述できることが重要であり、さらに、システム全体の記述が複雑にならないように部品間の依存関係を記述できなければならない。

2.3　ペトリネットによる記述法

上記の問題を解決するために、GUIのふるまいをペトリネットを用いて記述する方法を検討する。あるトランジションに対する入力プレースは、トランジションが発火するための条件となるので、GUIの反応の条件となる「部品の状態」とする。トランジションは、条件が成立した時に発火する事象であり、GUIの反応とする。出力プレースは、GUIが反応した結果の、部品の状態を示す。プレースとトランジション間のアローは、これらの因果関係を示す。さらに、発火するトランジションがない場合に、トークンを消滅させるプレース◎を導入し、これをユーザからの入力の表現に用いる。

Description of Behavior for Graphical User Interface with Petri Net

Koichiro UENO, Naoki TAMURA, Tsuyoshi NAKAJIMA, Kenji UEHARA

MITSUBISHI ELECTRIC Corp.

3　記述実験

3.1　記述の対象と結果

例題として、X-Windowのアプリケーションであるxcalendar(図１)のふるまいを記述した。ユーザは、各画面を互いに並行に操作できる。これらの画面に注目し、ペトリネットで記述した結果が図２である。図２において、○は画面の表示状態、◎は対応するボタンの操作に対応している。ユーザがボタンをクリックすると、対応する◎上にトークンが現れる。

3.2　考察

図２のペトリネットによる記述と比較するために、同じ例題を状態遷移図を用いて記述した結果を図３に示す。図３の状態遷移図では、ユーザの１つの操作が複数のアローとして（例えば、「quitボタンのクリック」は４本のアローとして）表される。従って、ユーザのある操作に対する反応を知るためには、すべてのアローを調べる必要がある。一方、図２のペトリネットでは、ユーザの操作に１対１で対応するプレースからアークをたどることにより、調べることができる。このことは、図の記述にも影響を与えている。状態遷移図ではステートの記述後に、それらを結んでユーザの操作が記述される。しかしペトリネットでは、ユーザの操作を必要に応じて記述できる。従って、ペトリネットは状態遷移図に比較して、ユーザの操作中心に記述することが可能となる。

4　おわりに

本報告では、GUIのふるまいをペトリネットで記述する方法を提案した。また、実際に記述実験を行なった結果、ペトリネットを用いることによりユーザの操作中心の記述が可能なことが確認できた。今後の課題として、ペトリネットによるふるまいの記述と、部品の配置による見ための記述の関係付けの方法を検討したい。

【参考文献】

[1]"マルチメディア時代のユーザインタフェース", 日経コンピュータ別冊ソフトウェア, 日経BP社 (1989-7).

[2] D. Harel, et al., "STATEMATE:Working Environment for the Development of Complex Systems", Proc. of the 10th IEEE ICSE , (1988-4).

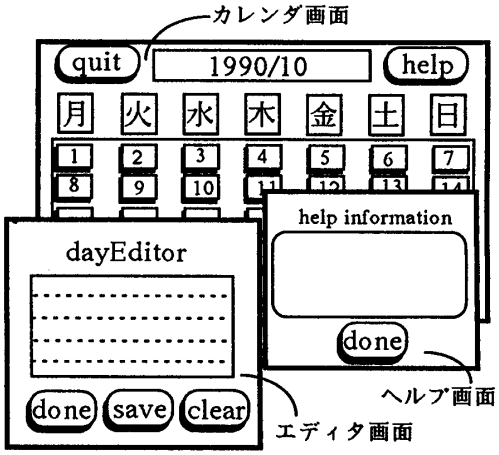
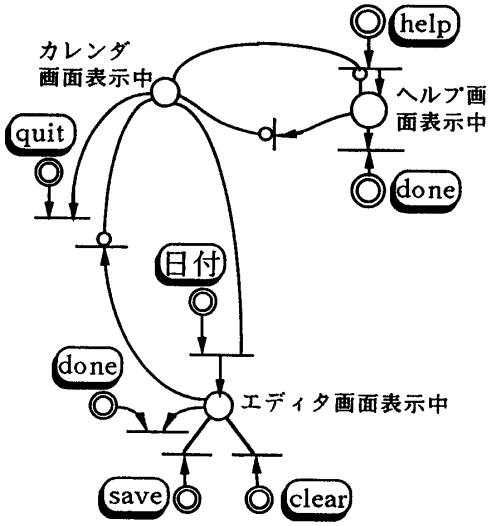


図１ xcalendarの見ため



記法上の拡張

: 入力プレースにトークンがあれば、出力トランジションを発火できなくするアーク

: トランジションが発火しても、プレースからトークンを流さないアーク

図２ ペトリネットによるふるまいの記述

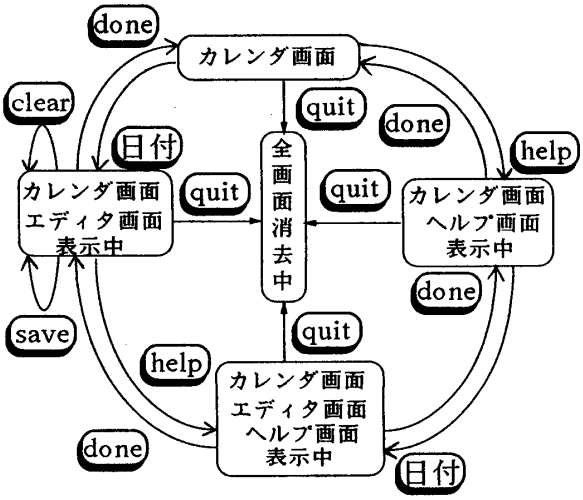


図３ 状態遷移図によるふるまいの記述