オブジェクトフレームワークを用いたカードゲームプログラムの作成支援

1J－6

松永 賢次
専修大学経営学部

1 背景

カードゲームやボードゲームといった思考ゲームのプログラム開発について考える。作成するプログラムは，1人以上のn人プレーヤーが参加し，コンピュータプログラムによるプレーヤーまたは，プログラムのインタフェースを通して人間がゲームに参加するものとする。その場合，プログラムは，

1. コンピュータプログラムによるプレーヤー
2. 人間がゲームに加わるためのインタフェース

の2つの部分から構成されると考えられる。プログラム開発において，次のような問題が存在している。

プレーヤープログラムについて

- 仕様が最後まで確定しない。ゲームの規定についてはあらかじめ確定しているが，プレーの戦術戦略についてはプログラムの開発過程において常に修正がなされる。
- 探索スピードと変更の容易さが相容れない。プレーの強さは，どこまで探索するかに依存している。プログラムコードは高速化をはかるため，ゲームの規定・戦略・戦術を複雑に変換したものになるので，追加や変更は容易でない。

インタフェースについて

- プレーヤープログラムと同じゲーム規定に基づいて作成されなければならないが，一致させることが容易ではない。
- 実世界のゲームと同じようなインタフェースを作成するためには，ライブラリーとその利用法を用意する必要がある。

本研究では，思考ゲームプログラム開発における以上述べてきた問題点を解決するために，人間が行っていた部分を自動化，あるいは人間に対してより良い支援を与えることを目標としている。

2 基本的なアイディア

再利用のためのシステム構成

思考ゲームにおいては，着手を決めるためのアルゴリズム，ゲームに登場するオブジェクト，ゲームのルールなど共通のものが多い。そのため，新しいゲームに対するプログラムを開発する際に，既存のプログラムを再利用することが有益である。

再利用のための技術として，オブジェクトフレームワーク[1]の考え方が有効であると考えている。全体の制御について再利用し，そこから呼び出される一部の関数についてカスタマイズするものである。思考ゲームにおいて用いる探索アルゴリズムは，α－β法，A*アルゴリズムなど一般的なものであり，個々のゲームの状態表現と状態の変更を行うオペレータをカスタマイズすればよい。

また，思考ゲームにおいては，ゲームに登場するオブジェクトは見た目，機能ともに再利用することが可能であり，クラスライブラリとして用意することができる。

仕様の記述

個々のゲームを記述する言語は，次のような性格

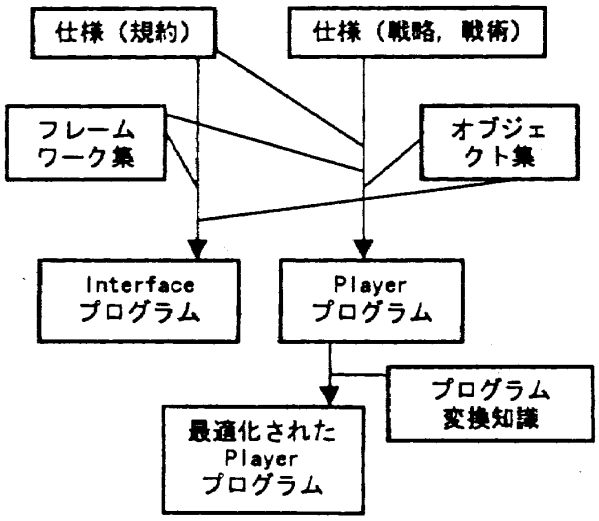

図1：思考ゲーム作成の流れ

Developing Card Game Programs Using Object Framework
Kenji Matsunaga
Senshu University
2-1-1, Higashi-mita, Tama-ku, Kawasaki, Kanagawa
214-8580 Japan

を持つべきであると考えている。

1. 全体の制御については用意されたオブジェクトフレームワークを（直接，あるいは間接的に）利用できること。
2. フレームワークに対してカスタマイズすべき部分は，宣言的，あるいは断片的な手続きで記述できること。
3. ゲームに登場するオブジェクトも（直接，あるいは間接的に）利用できること。

また仕様記述は，次の部分から構成される。

- プレーヤープログラムとインタフェースプログラムとで共通の仕様（ゲームの規定）
- プレーヤープログラムのみに関係する仕様（ゲームの戦略や戦術）とインタフェースプログラムのみに関係する仕様(GUI)

3 例

ここでは，カードゲーム Calculator を例にとり，説明していく。

ゲームの規約は次のとおりである。一人ゲームであり，プレーヤーは Joker を除く 52 枚のカードを持ち，4 つのカード列を作成していく。カード列は，それぞれ番号 1, 2, 3, 4 からスタートさせ，前のカードに対してそれぞれ番号が 1, 2, 3, 4 ずつ多いカードを加えていく。別にスタックをいくつか（通常は 3 または 4）持ち，手札からスタックに，あるいはスタックからカード列に対してカードを動かすことができる。すべてのカードが 4 つのカード列に動いたら成功であり，カードをカード列に動かすことができなくなったら失敗である。

```
(UseFramework 'OnePlayerCardGame)
(InitialState
  (setq stack1 (make-instance 'Stack))
  .....
  (setq queue1 (make-instance 'Queue))
  ....
  (setq hand (make-instance 'Queue))
  (setq deck (make-instance 'CardDeck))
  (allCardsWithoutJoker (deck))
 )
(SucceedCondition
   (and (empty deck) (empty hand)))
(FailCondition
 .....)
(Operator
  (moveCard from: top(deck) to: last(hand)
    precondition: (and (empty hand) (not (empoty deck))))
  (moveCard from: top(hand) to: last(stack1)
    precondition: (and (empty hand) (not (empoty stack1))
      (or (= (number(top hand)) 1)
          (= (number(top hand))
             (+ 1 (mod (- (number(last(stack1)) 1) 13)))))
  .....
)
```

図 2：Calculator の仕様記述の一部

仕様記述言語はこれをできるだけ直接的に記述できることが望まれる。カードゲームの場合は，何人ゲームなのか，複数人のゲームであれば順番，必要なカードの種類，ゲームに登場するカード列とその可視性・挿入や取り出しの方法，プレーヤーが取り得る行動，ゲームの勝ち負けや終了の方法，を記述すれば良い。図 2 に一部を載せてあるが，構文解析の簡便さのため Lisp のような記述法を用いている。

この記述から，適当なフレームワーク（この場合は 1 人用のカードゲームのインタフェースとプレーヤー）を選択し，さらにそこに埋め込むプログラム（Java によるもの）を生成する。現在は，戦術や戦略についてまだ記述できておらず，また高速化のためのプログラム変換には手がついていない。

4 今後の課題

基本的なアイディアに基づいて，Calculator の規約からプログラムを生成するところまでできたが，まだ多くの課題が残っている。1 つは，オブジェクトフレームワークやライブラリーの記述方法についてである。2 つ目として，プレーヤープログラムの効率の良いプログラムへの変換方法についての研究をしなければならない。

今回提案した枠組みを使えば，思考ゲームのトレーニング戦略の記述方法と変換ルールを用意することで，トレーニングソフトウェアに発展できると考えているので検討してみたい。

参考文献

[1] Brown, A.W. (ed.), Component-based Software Engineering, IEEE, 1996.