Flageアーキテクチャのためのプログラム合成メカニズム*

4N－9

松浦佐江子†　本位田真一‡
新ソフトウェア構造化モデル研究本部
情報処理振興事業協会（ＩＰＡ）

1　はじめに

われわれは環境の変化に柔軟に対応する協調型ソフトウェアアーキテクチャ（Flageアーキテクチャ）のためのプログラム合成メカニズムについて研究している[2]。本メカニズムはプログラムの作成プロセス・仕様の変更プロセス・プログラムの変更プロセス・変更方法をまねるプロセス等、プログラム開発において人間が活用するさまざまな経験的知識の蓄積と利用方法を提案するものである。本稿では環境の変化をプログラムの外部環境である仕様が変更された場合と考え、これらの知識を用いて既存のプログラムを仕様変更要求を満たすように変更する方法について述べる。

2　経験の利用

Flageアーキテクチャにおけるエージェントの行動の1つは仕様が変更された時に自己のもつプログラムを修正することである。このような行動を実現するために、エージェントが自分の経験や他エージェントの経験を利用することに着目した。エージェントの利用できる経験として、まず挙げられるのは元のプログラムをどのようにして作成したかという経験であり、これが仕様変更要求に適合したプログラムを作成するための基礎知識となる。

経験を利用するという人間の自然な行為をソフトウェアの世界で実現するためには、つぎの2点が前提となる。

(1) エージェントは自分の経験をもっている。すなわち、仕様からプログラムを作成したプロセスを獲得できる。

(2) エージェントは環境の変化を認知できる。すなわち、仕様変更のプロセスを獲得できる。

これらの前提のもとでエージェントが自己の経験を利用するメカニズムとしてつぎの2つの方法を提案する。

(3) プログラムの作成経験をまねて変化に適合する。

(4) まねた経験をまねて変化に適合する。

本稿ではエージェントの経験を**自己形成プロセス**と呼ぶ。前提および2つの方法を自己形成プロセスを利用したまねるメカニズムとして、つぎのように実現する（図1参照）。

(1) 述語論理によって仕様を定義し、自然演繹による推論を行ない、その結果をλ式に対応付ける。さらにプログラム変換技術の応用によって目的のプログラムを作成する。このプロセスを**合成プロセス**と呼ぶ。

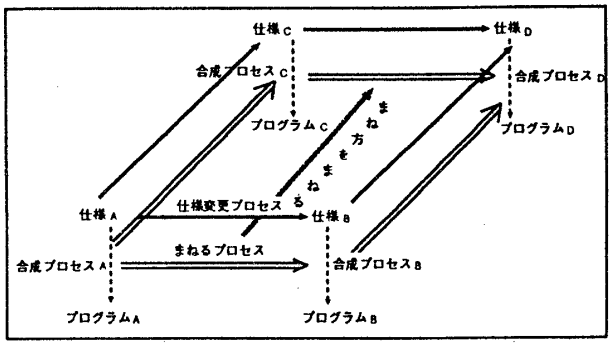

図 1: エージェントの経験

(2) 型・論理式の変化を定式化し、**仕様変更プロセス**を獲得する関数群を定義する。これらの関数を**モジュール操作関数**と呼ぶ。

(3) (2) のプロセスを利用して合成プロセスを修正する関数群および手順を定義する。これらの関数を**プロセス操作関数**と呼び、このプロセスを**まねるプロセス**と呼ぶ。

(4) (3) の経験を**まね方**と呼び、これを修正して利用する条件および手順を定義する。このプロセスを**まね方をまねるプロセス**と呼ぶ。

一方、他エージェントの自己形成プロセスを利用するためには自分の世界にない経験を自分流に変化させて利用することが必要である。すなわち、エージェントは自分と他人の関係を認識できなければならない。この問題は5節で検討する。

3　場の設計

システムに対する要求は一時に全てが提出されるわけではない。むしろ、要求は五月雨式に設計者に与えられる。すなわち仕様変更要求のシステムへの反映は線形ではなく並行に行なわれると考えるのが自然である。例えば、自動車の走行シミュレーションを考える。自動車が走行するさまざまな環境（交差点・2車線道路・対面交通等）における「走行」の仕様を考えることができる。これらの要求は全て「走行」に対する要求であるが互いに独立して解決することができる。また自動車は一定の環境ではなく適宜複数の環境の重なりに入り、その環境に応じて走行する。そこで、システムを基本的な要求を満たす核となる部分からつくりはじめ、並行開発の可能な要求に対してはそれぞれ独自に変更を行なうことは自然であり、このような開発によってエージェントは再利用しやすい単位の経験を蓄積することができる。図1に示したように、核になる仕様A（交差点での走行）をまねて仕様B（2車線交差点での走行）を開発したり、同じ仕様Aを別の観

*Program Synthesis Mechanism for Flage Architecture: Saeko Matsuura, Shinichi Honiden
†管理工学研究所より出向
‡現（株）東芝

点でまねて仕様C（対面交通時の走行）を開発した結果、各々の開発経験であるまね方をまねて2車線交差点における対面交通時の走行プログラムを開発することができる。図1における仕様・合成プロセス・プログラムがFlageアーキテクチャにおける「場」に蓄積される。エージェントは各場において合成されたプログラムによって行動する。

4 まねるメカニズム

現在、Extended ML[3]を仕様記述言語、合成されるプログラムを関数型言語MLとしてまねるメカニズムのプロトタイプシステムを開発中である。合成プロセスは自然演繹法における推論規則の公理への適用・論理式とλ式の対応付け・共用変数の導入等のプログラム変換規則の適用により構成されるプロセスである。モジュール操作はデータ変換・公理の追加や削除・公理前提部への論理式の追加等のML記述のモジュール操作関数の組合せで定義される。一方、プロセス操作は合成プロセスを変更する手段であり、モジュール操作関数の適用・推論規則等の適用・合成プロセスのサブプロセスの操作といったML記述のプロセス操作関数の組合せで定義される。各操作関数の入力は、対象を各々の形式で構造化して表現したMLのデータである。

仕様変更はモジュール操作を定義した新しい場の生成によって表される。エージェントは合成プロセスを携えて新たに生成された場に入る。そして変更された仕様を獲得し、まねるメカニズムを使って合成プロセスをまね、変更された仕様を満たすプログラムを合成する。この時モジュール操作やプロセス操作の関数を仕様や合成プロセスに順次適用するプロセスがエージェントの経験として場に蓄積される。すなわちモジュール操作やプロセス操作により蓄積された経験がまね方である。まね方は仕様や合成プロセスに含まれる項を入力とした関数の列であり、エージェントの経験として利用することができる。一方、まね方を携えて場に入ることによって、まね方をまねてプログラムを合成することができる。

5 他エージェントの経験

Flageアーキテクチャにおけるエージェントの特徴の1つは場を渡り歩いて必要な機能や情報を獲得することである[1]。本稿で述べたまねるメカニズムは、エージェントがプログラムを作成する過程で獲得した情報を利用するメカニズムであり、自己形成プロセスを場に蓄積することによって自己の経験を利用したエージェントの環境の変化への適合が達成される。

一方他エージェントの経験は利用できるだろうか。例えば、3節で述べたように日本における交通規則に従って自動車が走行する場を日本ノードにおいて設計したとする。日本車エージェントは日本ノード内の場を渡り歩いて走行のプログラムを拡張した。ここにアメリカ車エージェントがやってきて走行プログラムを獲得する場合を考えてみよう。図2中の番号は以下の番号と対応する。

① アメリカ車エージェントは日本とアメリカの交通規則の差異を知っており、規則を表す公理の置き換えをモジュール操作関数によって定義できる。これが他人との関係を認識する1つの例である。まず日本ノードにおける核となる場（交差点）に入り、差異に基づき自己の仕様を合成し新たな場を生成する。図2における黒丸がこの仕様を、黒丸への矢線が場の移動を表す。

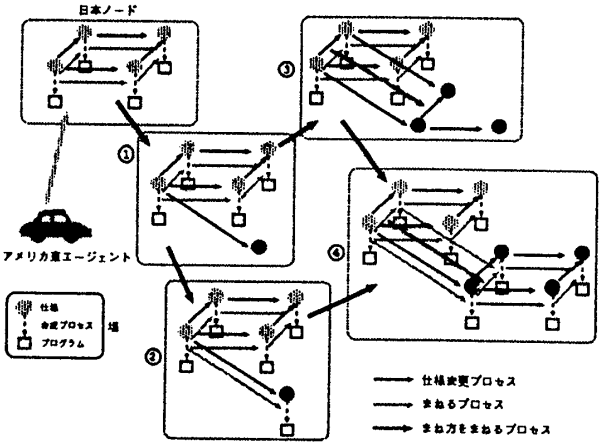

図 2: 他エージェントの経験の利用

② 交差点の場にある合成プロセスを持って新たな場に移動し、まねるメカニズムを用いて交差点におけるアメリカの交通規則に従った走行のプログラムを合成する。

③ 一方、アメリカ車の走行の仕様が日本ノードで構築されたさまざまな機能をもつように、仕様のまね方をまねて各機能に相当する仕様を定義する。アメリカ車の2車線道路での走行仕様や対面交通時の走行仕様が作られる。

④ まね方をまねる条件が成立しているところでは、合成プロセスのまね方をまねてプログラムを合成することができる。

6 おわりに

本稿ではエージェントが場を渡り歩いてプログラム開発経験を獲得することを目的とした場の設計方針およびまねるメカニズムによる経験の利用方法について述べた。仕様からプログラムを合成することをエージェントの経験の基礎としていることから、本メカニズムをFlage言語において実現することができれば、蓄積された経験を組み合わせて新たな環境にも動的に適合することが可能となるが、これは今後の課題とする。

謝辞 本研究は、産業科学技術研究開発制度「新ソフトウェア構造化モデルの研究開発」の一環として情報処理振興事業協会（IPA）が新エネルギー・産業技術総合開発機構から委託をうけて実施したものである。

参考文献

[1] 粂野、他：「Flageアーキテクチャの構想」，第51回情報処理全国大会, 1995.

[2] 松浦，本位田：仕様変更のプログラムへの写像 — 仕様変更プロセスを利用したプログラム合成 —，情報処理学会論文誌，Vol.36, No.5, pp.1211-1227.

[3] D.T.Sannella and A. Tarlecki:Program specification and development in Standard ML,in Proc.12th Ann.ACM Symp.on Principles of Programming Languages(1985),pp.67-77.