環境の変化に柔軟に適用できる「発生型ソフトウェア」の提案

1S−6

大林　正晴*　　本位田　真一**

情報処理振興事業協会（ＩＰＡ）

１．はじめに

ネットワーク通信プロトコル、開放型の並列分散システム、ユーザインタフェースやマンマシン系、ソフトウェアプロセスなどのように自律的な実行主体が存在し相互に通信し合いながら並行的に動作するシステムを計算機で、つまりソフトウェアで自由に扱えるようにしたいと言う要請が高まっている。また、従来のソフトウェアは、機能的には固定的なものであり修正や変更などに対して必ずしも柔軟なものではない。

筆者らは、生体情報系に着目し環境の変化に適応する仕組みをソフトウェア部品に応用することを目指している。特に、遺伝子を生体の発生や分化、恒常性の維持などの制御プログラムとみなし、その本質的な特質をソフトウェア部品としてモデル化することを試みている［１］［２］［３］。

最終的な目標は、要素間の協調動作の結果として意図した仕事を行わせるようなソフトウェアの新しい仕組みを確立することである。

２．従来のオブジェクト指向

まず、従来のオブジェクト指向プログラミングの特徴について、ソフトウェア部品の観点から考察してみることにする。

（クラス定義の階層構造）

オブジェクト指向プログラミングの特徴の１つとして、階層構造を用いたクラス定義を上げることができる。上位クラスの属性やメソッドなどを下位クラスが継承することにより、いわゆる差分プログラミングが可能で、コード量を圧縮する強力な手段になっている。

（インスタンスの振る舞いの構造）

クラス定義の階層構造以外に、インスタンスレベルの構造、つまり、オブジェクト相互参照構造を持っている。具体的には、クラス定義から生成されたオブジェクトがどのようにメッセージをやり取りし、目的の機能を果すか、そのようす（振る舞い）のことである。このインスタンス相互参照構造が、問題解決のためのアプリケーションを反映したものになっている。

一般にオブジェクト指向プログラミングの長所として、つぎの３点を挙げることができる。

①対象モデルを系統的に構成できる
②記述を簡潔にすることができる。
③部品としての再利用の機会が増える。

しかし、このようなオブジェクト指向プログラミングは、ソフトウェア開発における諸問題を根本的に解決するものではない。短所もいくつかある。

クラス定義から生成されるオブジェクトの振る舞いを決定するメッセージの送り先は、そのクラス定義の中に記述されており、普通はインスタンス変数などで表わされたオブジェクトに固定されている。アプリケーションの機能変更や拡張などで、それらを変える為には当然クラス定義を変更しなければならない。

一方、クラス定義は、一般に系統的に作られており上位のクラスは、多くの下位クラスをもつことになる。そのようクラス階層で、上位のクラス定義の変更は多くの下位クラスに影響を及ぼすことになる。もちろん、場合によってはそのような上位クラスの変更が下位クラス全体に有意義なこともあるが、必ずしもそうでないことも多い。

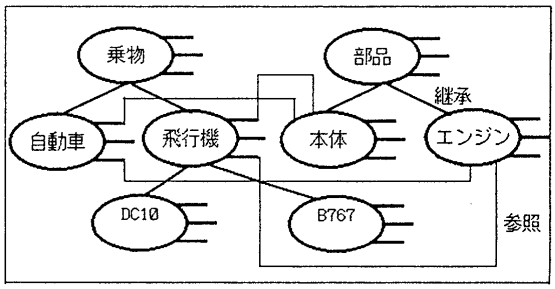

図１．オブジェクト定義の構造例

このようにクラス定義の階層構造が厳密に決められているので変更が難しくなる。これを解決する１つの方法は、クラス定義の階層を深くせずに、なるべく平坦にすることである（差分プログラミングの利点は失われるが）。

クラス定義を部品化の仕組みとしてみたとき、情報隠蔽がなされているので参照するインスタンスの内部構造や実現には影響されず、ある程度自由にシステムの中に他の部品を組込むことが可能である。しかし、実際には、メッセージのインタフェースや機能などを厳格に合せなかればならないなど制約も多く、再利用を難しくしている。

この問題に対しては、いわゆるリフレクション機構などをもちいてクラス定義を動的に変更して解決しようとする研究も行なわれている。筆者らは、これらの問題を解決するために、「発生型ソフトウェア」という新しい概念に基づく枠組みを提案している。

３．発生型ソフトウェア

ここでは、個（エージェント）とよばれる複数の自律的な要素があり、それらの個の間および環境との相互作用を通じて秩序を形成し、特定の機能を果すとともに環境の変化に対して適用できる能力を備えたものをエージェントモデルとして考える。

３．１　定義

（１）ソフトウェアプロセス

一般的には、ソフトウェア開発の諸過程をソフトウェアプロセスと呼び、種々の研究がなされている。現実のソフトウェアプロセスは、採用する方法論や開発チームなどの人間的要素を含んだものであり複雑である。特に、各プロセスは、単純に進行するのではなく、バックトラックなども頻繁に起る。

ここでは、エージェントモデルのためのシステム構築の論理、つまり、エージェントの組合わせの論理過程のことをソフトウ

A Proposed New Framework of Software including Software Processes
Masaharu OBAYASHI, Shinichi HONIDEN
Information-Technology Promotion Agency, Japan
*㈱管理工学研究所から出向、**㈱東芝から出向

ェアプロセスと定義する。
具体的には、最終成果物（エージェントの集合）を得るまでに起った中間の成果物の変化のみに着目し、バックトラックなどの過程は基本的には含まれないものとする。
（２）発生型ソフトウェア
秩序の形成と環境変化への適用を実現するために、新しい考え方「発生型ソフトウェア」を提案する。すなわち、システムの組み立ての論理（ソフトウェアプロセス）をも内蔵したソフトウェアのことを発生型ソフトウェアと呼ぶことにする。
我々のエージェントモデルでは、エージェント相互の参照構造を順次、組み立てるような論理を自らのエージェント定義の集合の中に含むような体系を考える。つまり、個々のエージェント定義の中ではなく、別にエージェントの組合わせ方を制御するプログラムが、同じエージェントとして定義されているのである。

３．２　環境と世代
つぎに、発生型ソフトウェアにおける環境と世代の概念について説明する。ソフトウェアプロセス（ＳＰと略す）と対象とする問題領域を表わすアプリケーション（ＡＰと略す）とを区別して考える。
具現化されているエージェントの集合を環境と呼ぶことにする。また、発生したソフトウェアを初期環境ごとに世代と呼ぶ。
環境には、ＡＰを処理するエージェントだけでなく、ＳＰに関与するエージェントが共存する。これらの関係を示したものが図２である。ＳＰとＡＰの共通部分にあるエージェントは、各ＡＰに固有のＳＰを表わすエージェントである。
発生型ソフトウェアは、環境内のエージェント定義に従ってＡＰを段階的に形成して行く、環境を変化させることにより、環境に適応した類似のシステムを発生することができる。異なる初期環境のもとで発生したソフトウェアは、互いに世代が異なる。同じ初期環境のもとで発生したソフトウェアは同じ世代に属する。
ＳＰの機能強化やＡＰの機能強化など世代の変更は、人が介入して新たなエージェント定義を環境に追加したり、修正して行なう。

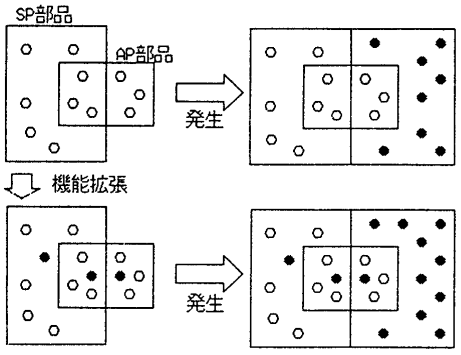

図２．発生型ソフトウェアの世代

３．３　部品
１つのエージェントの定義は、１つのソフトウェア部品とみなすことができる。初期環境には、システムを発生し、ＡＰを解くために必要な部品の集合が収められている。発生の過程で、組み立てられた部品が環境に追加され、高度な機能をもつエージェントとして振る舞う。
部品を分類すると、つぎのようになる。
①ＳＰ部品：　部品を組み立てるための汎用的な部品
②ＡＰ固有のＳＰ部品：
部品を組み立てるためのＡＰ固有の部品
③ＡＰ部品：　ＡＰを解くための部品

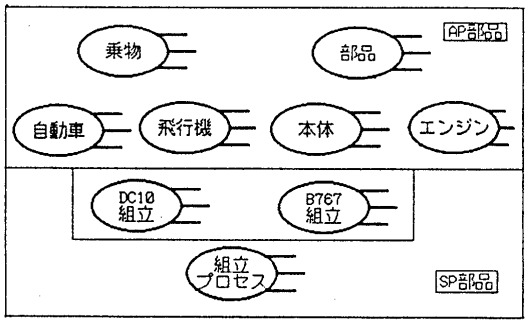

図３．発生型ソフトウェアの構造例

３．４　部品の要件

各部品は、自己完結的に定義ができる。すなわち各部品は、それぞれのエージェントの振る舞いを記述したものであり、自己制御の機能をもつ。また、部品間の相互作用は、環境に依存して決定される。すなわち部品と部品の結合の相手は、事前には設定されてはいない。実際に、ある部品がどの部品と結合するかは非決定的である。環境の中にセットされる部品の組合わせ方によって動的に決まる。
このような部品と部品および部品と環境の相互作用を通じて、システムの意図（大局的目標）を反映した秩序が形成されるように全体を制御しなければならない。また、環境を構成する部品を動的に更新するようなことも考える。
ここで提案した「発生型ソフトウェア」は、このよな要件を満たし、環境の変化にも柔軟に適応できる仕組みをもつ。

４．おわりに

現在、発生型ソフトウェアを定式化するために、プロセス代数の１つである化学抽象マシン（Chemical Abstract Machine）を用いて、例題の記述を試みている［３］。
特に、部品の接合形式や相互作用による調節の仕組み（抑制と促進）の枠組みが重要であると考えている。

［謝辞］
本研究は、次世代産業基盤技術研究開発「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が新エネルギー・産業技術総合開発機構から委託されて実施したものである。

参考文献
［１］本位田真一「協調アーキテクチャによるソフトウェアの自動生成」人工知能学会誌、Vol.6 No.2, pp184-186(1991)
［２］大林正晴、本位田真一「生体情報系における協調システムについて」情報処理学会第４３回全国大会、3K-9、1991
［３］大林正晴、本位田真一「協調エージェントのプロセス代数による定式化－「発生型ソフトウェア」の提案」ｿﾌﾄｳｪｱ工学研究報告、情処研報 Vol.92, No.38