

構文解析を用いたオブジェクト指向分析手法

3 G-1

稲葉稔智 笑輪行真 石原平太郎

沖電気工業株式会社

1. はじめに

オブジェクト指向によってソフトウェアを開発するフェーズは大きく2つに分けられる。1つは分析フェーズであり、もう1つは設計フェーズである。分析フェーズでは、問題を解決するために必要となるオブジェクトを抽出し、設計フェーズでは、分析フェーズで得られたオブジェクトを実際に設計する。また、オブジェクト指向によるソフトウェア開発手法について言及した論文もいくつかある。^{1) 2) 3)} いずれの手法も問題の分析にはSA手法⁴⁾を用い、その出力であるDFDからオブジェクトを抽出している。しかし、この方法ではDFD中に現われるデータとそれを取り扱うプロセス間の結合、すなわちオブジェクトとそれが持つべき機能の結合を決定する明確な指針がないし、これは設計者の感覚によるところが大きい。このことは、問題の解決のために、必ずしも適切なオブジェクトが選ばれるとは限らないことを意味する。そこで今回は、SA手法によらない分析手法、および分析手法の核となるオブジェクトの抽出技法について報告する。

2. オブジェクトの抽出技法

オブジェクト指向分析では、問題を解決するために必要となるオブジェクトの抽出に主眼をおく。ここでは、オブジェクト指向分析の核となるオブジェクトの抽出に、構文解析を適用する技法について述べる。

2. 1. 5つの基本構文

実世界での我々の言動はすべて以下の5構文を使って表現できる。

- 第1構文 (S+Vi)
- 第2構文 (S+Vi+C)
- 第3構文 (S+Vt+O)
- 第4構文 (S+Vt+Oi+Od)
- 第5構文 (S+Vt+O+C)

ここで、

- S : 主語
- Vi : 自動詞
- Vt : 他動詞
- O : 目的語
- Oi : 直接目的語
- Od : 間接目的語
- C : 補語

2. 2. 構文解析によるオブジェクト抽出技法

2. 1項で述べた5構文それぞれについて、オブジェクトおよびオブジェクトが持つべき機能(メソッド)の抽出方法を以下に示す。ただし、obはオブジェクトを意味し、mdはメソッドを意味する。

(1) 第1構文 (S+Vi)

例) The car moves.

解析結果) The car, move.
ob(S) md(Vi)

これはSはViすると解釈される。この構文では、Sがオブジェクトであり、ViがSのメソッドになる。

(2) 第2構文 (S+Vi+C)

例) John comes running.

解析結果) John, come running.
ob(S) md(Vi+C)

これはSはCとしてViであると解釈される。この構文では、Sがオブジェクトであり、Vi+CがSのメソッドになる。

(3) 第3構文 (S+Vt+O)

例) The cat kills a rat.

解析結果) The cat, kill a rat.
ob1(S) md1(Vt) ob2(O)

これはSはOをVtすると解釈される。S、Oがオブジェクトであり、VtがSのメソッドになる。

(4) 第4構文 (S+Vt+Oi+Od)

例) John buys Marry a toy.

解析結果) John, buy a toy.
ob1(S) md1(Vt1) ob3(Od)
Marry, own the toy.
ob2(Oi) md2(Vt2) ob3(Od)

これはSはOiにOdをVtすると解釈される。この構文では、S、Oi、Odがオブジェクトである。メソッドVtは基本的にSのメソッドとして取り扱うが、SのメソッドVt1とOiのメソッドVt2に分解される場合もある。

(5) 第5構文 (S+Vt+O+C)

例) Marry helps her mother cook.

解析結果) Marry, help her mother cook.
ob1(S) md1(Vt) ob2(O) md2(C)

これはSはOをCとしてVtすると解釈される。この構文では、S、Oがオブジェクトであり、VtがSのメソッドである。ただし、Cの部分がOのメソッドになる場合もあり得る。

3. オブジェクト指向分析

ここでは、オブジェクト指向によって問題を分析するための基本的な考え方、オブジェクトの抽出を完了するまでの手順、および分析フェーズで使用するツールについて述べる。

3. 1. 分析手法

通常の我々の言動は、すべて2節で述べた5つの基本構文を使って表現することができる。したがって、オブジェクト指向で問題を解決する場合、問題の分析をそのような基本構文を使って行なうことが自然であると考えられる。

問題の分析は、2節で述べた5構文を使って、設計しようとしているソフトウェアを第3構文で表わす("What")ところから始まる。次に、その構文のVtを実現する方法("How")を考える。さらに、2節で述べたオブジェクト抽出技法によって"How"中に発見したオブジェクトに対して"What"を適用し、また"How"を適用することを繰り返す。そして、これ以上オブジェクトが出現しなくなったところで分析は終了する。

3. 2. 分析手順

分析フェーズは以下の手順で進められる。

(1) 問題のオブジェクト指向分析

これは3. 1項で述べた手法にしたがって行なう。この工程を終了した時点で、問題を解決するために最低限必要となるオブジェクト、およびその機能を明確にすることができる。

(2) オブジェクトの階層化

Object Oriented Analysis by Parsing

Toshinori INABA, Yukimasa MINOWA, Heitaro ISHIHARA

Oki Electric Industry Co., Ltd.

階層化すべきオブジェクトというのは、同じ様な機能を持つオブジェクトである。これより、問題の分析から得られたオブジェクトの中で、1つでも同じメソッドを持つオブジェクトを階層化の対象とする。

(3) オブジェクトフローを使った分析結果の記述

これは分析の結果得られたオブジェクトを使って、問題の解法をオブジェクト同士のメッセージパッシングによって表わす。これによって、分析では得られなかったオブジェクトの機能を見つけたことができる。例えば、問題を解決するための一連の流れの中で、1つのオブジェクトから、他のオブジェクトへデータを渡すためだけに必要となる機能などが考えられる。

(4) オブジェクトの整理

これは(1)から(3)の工程を通して得られたオブジェクト、およびその機能、またオブジェクトの階層を、設計フェーズでの工程のために整理するものである。

3. 3. 分析ツール

ここではオブジェクト指向分析フェーズで使用する各種のツールについて述べる。

(1) 分析記述ツール

これはオブジェクト指向分析を進めていく途中で発見されたオブジェクト、およびその機能、さらに全体の流れを記述するために使用する。これには以下のようなものがある。

- ・ [] : 分析中に発見されたオブジェクトを表わす。
- ・ { } : オブジェクトのメソッドの開始と終了を表わす。
- ・ () : 処理を表わす。
- ・ *** : 繰り返すを表わす。

また、上記ツールを使った分析例を以下に示す。

[ファイルリーダー] ファイルリーダーは、与えられたファイルを読む。

1. (ファイルを開く。)
2. [ラインリーダー] ラインリーダーは、与えられたファイルを1ライン読む。
*** ファイルエンドまで ***
3. (ファイルをクローズする。)

(2) オブジェクトフロー図

これは分析の結果得られたオブジェクトを使って、問題の解法を記述するために使用する。以下にその例を示す。

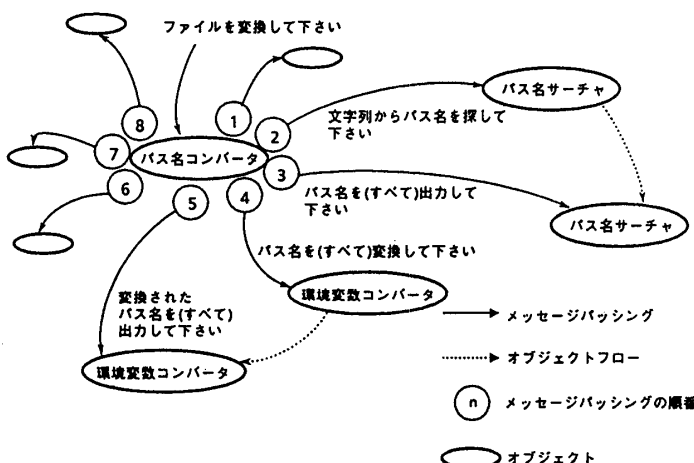


図1. オブジェクトフロー図

4. 分析手法の考察

2節、3節において問題のオブジェクト指向分析手法について述べた。ここでは、その手法にしたがって実際にツールを試作した結果からその手法を考察する。

4. 1. 試作ツール

VAX/ULTRIX¹⁾上で動作し、VAX/VMS²⁾フォーマットのCソース、アセンブル、リンカオプション各ファイルをもつVAX/ULTRIXフォーマットに変換するツールを試作した。またツールの仕様記述は以下の通りである。

本ツールは、VAX/VMSフォーマットのファイルを読み込み、そのファイル中にパス部分を見つけた場合、その部分をVAX/ULTRIXフォーマットに変換し、拡張子 .pcv をつけて出力する。変換は被変換部と同一の名前を持つ個人環境変数が設定されている場合に限って行い、設定されていない場合はエラーメッセージを通告する。

4. 2. ツールの分析

4. 1項で述べたツールの仕様記述にオブジェクト指向分析を適用し得られたオブジェクト、およびその継承関係は以下の通りである。

```

[ラインリーダー] → [ファイルリーダー]
[環境変数リーダー]
[カラムライター] → [ラインライター]
                                ↳ [ファイルライター]
[サーチャ] → [Cソースパス名サーチャ]
                                ↳ [アセンブルパス名サーチャ]
                                ↳ [リンカパス名サーチャ]
[コンバータ] → [環境変数コンバータ]
[コンバータ] → [環境変数コンバータ]
[パス名コンバータ]
[変換規則] → [変換規則表]
[文字列] → [文字列表]

```

ただし、

記号→は継承関係を表わし(例: 親→子)、記号[]はオブジェクトを表わす。

4. 3. 考察

今回の試作は、はじめにCソースファイルのみを変換するツールを作成し、次にアセンブルファイル、リンカオプションファイルを取り扱えるように機能追加するという形で行なった。機能追加は、Cソースファイル用パス名コンバータのオブジェクトフロー図から、サーチャ部分の変更を行なうだけでよいことがわかった。そしてアセンブルファイル、リンカオプションファイル用のサーチャを、Cソースファイル用から継承させた。このことから、2節、3節で提案した分析手法は、問題を解決するためのオブジェクトをかなり低いレベルまで見つけることができると考える。これはまた、オブジェクト指向で作成されたソフトウェアの特徴である部品化による生産性の向上を実現するものであると考える。

5. まとめ

本報告では、オブジェクト指向ソフトウェア開発の中で、特に分析フェーズについて述べた。そしてその手法を実際にツールを試作することで考察した。結果として、本手法はかなり低レベルのオブジェクトまで抽出することができ、オブジェクト指向の利点である部品化による生産性の向上を実現できることがわかった。今後は、より複雑な問題への本手法の適用可能性を考察したい。

参考文献

- 1) G. Booch: Object Oriented Development, IEEE Transaction on Software Engineering, February 1986.
- 2) B. Alabiso: Transformation of Data Analysis Model to Object Oriented Design, Proceedings of OOPSLA'88, San Diego, 1988.
- 3) 堀正弘, 他: 構造化分析を利用したオブジェクト指向設計の適用, 第40回情報処大4S-2, 1990
- 4) 高橋智弘, 黒田純一郎 監訳: 構造化分析とシステム仕様, 日経マグロウヒル社, 1986

¹⁾ VAX/ULTRIX, ²⁾ VAX/VMSはDEC社の商標です。