連載　オブジェクト指向と哲学
第 41 回　オブジェクト指向を分析する – オブジェクト指向の諸側面

河合 昭男
http://www1.u-netsurf.ne.jp/~Kawai

オブジェクト指向は本来プログラミング技術として世に出てきたものですが、その後上流系のモデリング技術にも取り込まれてきました。オブジェクト指向には様々な概念や用語が登場します。クラスとインスタンス／カプセル化／継承／多態性などの仕組みや属性／操作／メソッド／可視性などのクラスを構成する諸要素などオブジェクト指向を初めて学ぶ人には敷居が高いものです。

オブジェクト指向を外側から眺めて分析し、取り巻く様々な概念や用語から本質的なものと付随的なものを見極めてみたいと思います。

●プログラミング技術としてのオブジェクト指向

まずオブジェクト指向の原点から見てみましょう。プログラミング言語がオブジェクト指向言語であるためには、３つの条件が必要とされています。①カプセル化、②継承、③多態性が記述できることです。

① まずクラスの記述ができ、そこに属性と操作を一体化できることが前提です。次に、クラスを型としてその型の変数（オブジェクト）を定義し、そこに状態を持たせることができることです。さらに状態はオブジェクトの外部から直接アクセスすることはできず、公開操作を通してのみ可能とする仕組み（情報隠蔽）を持つことです。
② あるクラスを基底クラス（スーパークラス）とし、そのプロパティ（属性と関連）と操作を継承する派生クラス（サブクラス）が記述できることです。
③ 同じシグニチャ（操作名と引数の型）の操作が複数の異なる実装を持て、それらを実行時に選択できることです。

プログラミング技術としてこれらのオブジェクト指向の仕組みが取り込まれたのは当然ながら何らかの有用性があるからですが、本連載の趣旨はプログラミング技術そのものではなく、その中に何か世界の本質が隠されているのではないか、それは一体何かを考えることです。

次にモデリング技術としてのオブジェクト指向を見てみましょう。抽象度が一段上がったオブジェクト指向です。

●モデリング技術としてのオブジェクト指向

UML はオブジェクト指向に基づいたモデリングツールです。オブジェクト指向プログラミング言語にマッチしたモデルを記述できることを目指して策定されたものです。開発プロセスの各工程（要求→分析→設計→実装）により必要なモデルは異なります。

オブジェクト指向開発では最後の実装モデルを除いた３つのモデルに UML を使用することができます。実装モデルとはプログラミング言語で記述されたソースコードのことです。

システム開発で使用されるモデルは大きくは①機能モデル、②静的モデル、③動的モデルの３つの視点があります。UML には１３のダイアグラムが定義されていますが、各モデルで使用される主要なものを以下に示します。

① 機能モデルとしてはユースケース図とアクティビティ図が使用されます。UML の仕様書には確かにメタクラスとしてユースケースやアクティビティは定義されていますが、機能モデルに使用されるユースケース図やアクティビティ図はあまりオブジェクト指向っぽくはありません。

② 静的モデルとしてはクラス図とオブジェクト図が使用されます。やはり一番オブジェクト指向っぽいのはクラス図とオブジェクト図です。開発工程を通して様々なクラス図とオブジェクト図が用いられます。要求工程で使用されるクラス図やオブジェクト図は、問題領域で使用される概念を抽出してそれらの構造的関係を概念モデルとして表現することが目的です。クラスは抽象概念であり、オブジェクトは具体概念です。クラスの属性や操作などの内部構造は分析／設計工程で必要に応じてモデルを再構築しつつ追加してゆきます。

③ 動的モデルとしては相互作用図とステートマシン図が使用されます。相互作用図は４つありますが代表はシーケンス図です。シーケンス図はメッセージによるオブジェクトの協調動作を表すもので、クラス図と共にオブジェクト指向の最も重要なモデルを表現するものです。ステートマシン図はイベントによるオブジェクトの状態変化を表すもので、オブジェクトのライフサイクルを表します。

●概念モデル

人間には認識したものを概念化する能力が備わっています。五感、特に視覚で認識したものを具体概念とし、複数の具体概念から共通部分を発見して抽象化し抽象概念として整理します。更には複数の抽象概念から共通部分を抽出してより抽象度を高めた高位の抽象概念を階層的に作り出します。

認識できるものは自身の五感だけではありません。他の人が五感で獲得した概念を共有することもできます。さらには獲得した概念から新たな概念を演繹して作り出す能力も人には備わっています。

これらの概念はばらばらに孤立しているのでなく、なんらかのつながりがあり、それらを組み合わせた概念構造体と呼ぶべきものを人はそれぞれ保有しています。それらの概念構造体は人々

の間で一致している部分もあれば、一致していない部分もあります。
　この概念構造体はクラス図とオブジェクト図で可視化することができます。可視化することによりモデルを共有でき、あるいは各個人が理解しているモデルの差異を明確にすることもできます。

　概念にはその意味があります。意味を人はどのようにして理解し記憶するのでしょうか。

●辞書の説明方式

　「船を編む」（三浦しをん著）という小説は映画にもなり TV でも放映されましたが、我々が普段何気なく使っている辞書を作るのには強い意志と非常に長い年月が掛かっているのだということをあらためて教えられました。

　「右」を説明せよと言われても確かに困ります。子供の頃に箸は右手で持つ、逆に箸を持つ方が右手だと鶏と卵になりますが体で教え込まれます。だから言葉で説明できません。

　辞書は言葉の意味をどのように表現するのでしょう。ちなみに「右」は辞書には次のように説明されています。

　　広辞苑：南を向いた時、西にあたる方。
　　明鏡国語辞典：人体を対称線に沿って二分したとき、心臓のない方。

　「右」を説明するには広辞苑方式では「南」「西」の知識が前提となります。明鏡方式では「心臓」の知識が前提になります。これも奇妙な気がします。子供に右を教える前に東西南北または心臓を先に教えなければなりませんが、普通そのようなことはしません。辞書は体で教えることはできず、言葉でのみ説明しなければならないという制約があります。

　「東西南北」は広辞苑／明鏡共に次のような説明がされています。

　　「東」は日の昇る方角、「西」は東の反対方向、「北」は東の左方向、「南」は東の右方向

　東西は理解できそうですが、この説明では左右を知らないと南北は理解できません。広辞苑方式なら東西南北を知らないと左右は理解できません。辞書は言葉の言い換えや関係は記述でき、語彙を膨らましてゆくことはできますが、最低限の核となる語彙は前提として必要です。

●概念の内包と外延

　辞書は言葉の意味を他の言葉によって説明します。言葉とは概念に付けられた名前です。つまり辞書は概念の説明を他の概念で説明します。例えば「文房具」を調べてみましょう。

広辞苑：ペン・鉛筆・筆・紙・ノート・インク・定規など、物を書くのに必要なもの。文具。
明鏡国語辞典：物を書くときに必要な道具。筆・ペン・鉛筆・インク・紙・ノート・定規など。文具。

この説明方式はほぼ同じです。順序は違いますが、「物を書くための道具」という一般的な説明と「ペン・鉛筆・筆・紙... など」の具体例の列挙の２つから構成されています。前者を概念の内包（intension）、後者を外延（extension）と呼びます。このように概念は名前・内包・外延の３点セット（triad – ３つ組）で構成されています。「文具」は名前の別名です。
「文房具」の例では、辞書も概念の３点セットと同じ方式で説明しています。このような記述は自然な方法です。言葉や概念の説明には理解できる具体例が必要です。

--
今回はオブジェクト指向を分析するというテーマでプログラミング技術とモデリング技術の２つの側面を取っ掛かりとして始めました。次回もう少し考えを進めたいと思います。

以下次回。