GSLetterNeo Vol.124

2018 年 11 月

モデリングとは何か？

土屋 正人

はじめに

フローチャートから構造化手法、データ中心手法、オブジェクト指向分析設計と、対象領域をモデルで表現するやり方は、これまでソフトウエア開発において重要な位置にありました。しかしながら、近年、アジャイルな開発が広まる中、モデルを作ること――モデリングが軽視されているような気がします。アジャイルな開発ではモデリングは不要なのでしょうか？

今回はモデリングについて、改めて考えてみます。

モデリングとは

モデリングとは、言うまでもなくモデルを作ることです。モデルを作るということは、抽象化を行い、理解できる範囲を広げることにもなります。具体的なもの（インスタンス）の方が理解しやすいことがあるかもしれませんが、理解できる範囲が狭くなります。

モデルといっても、数式からチャート、ダイアグラム、シミュレーションなど多岐にわたりますが、ソフトウエア開発で使われるのは、主としてチャートやダイアグラムでしょう。

実存主義的モデリング

実存主義は 20 世紀フランスの哲学者・小説家サルトルに代表される思想ですが、「実存は本質に先立つ」という主張は、モデリングする上でヒントになることがあると思います。

ざっくりと記すなら、

- 「モノ」は本質が実存に先立つ
- 「ヒト」だけが実存が本質に先立つ存在

という点です。「実存は本質に先立つ」を「存在は役割に先立つ」と言い換えると理解しやすくなります。

「モノ」は役割があって存在があります。例えば、ペットボトルは飲み物を入れるという役割があって、存在します。

一方「ヒト」は存在があって役割があると言えます。

- 私という存在があって、（いま）講演者という役割を演じる
- 私という存在があって、（明日）受講者という役割を演じる

というように、役割は移り変わります。

では、データはどうでしょうか？　オブジェクトは？　クラスは？　システムは？　業務は？　ビジネスは？

データは単なる値です。適切な名前を付けることで「意味」を持ちます。適切な名前を付けることで「役割」を持つと言い換えることができます。

データの塊（例えばテーブル）も、適切な名前を付けることで「役割」が決まり、情報になります。

データとデータライフサイクルの塊（例えばオブジェクト）も、適切な名前を付けることで「役割」と「振る舞い」が決まります。

いずれも「存在があって役割がある」と言えます。存在に「名前を付ける」ことで「役割」を理解できるようになるわけで、名前を付けることができない、あるいは名前が適切ではないものは、共通理解が得られないものになります。

名前を付けることで役割を理解する

名前が適切ではない単純な例を挙げてみます。

例 1:

```
hoge = new Hoge()                    // hoge は何か？
hoge = hoge * 0.9                    // 何のためか？
                    //  再代入して副作用はないのか？
```

例 2:

```
anOrder = new Order()                // 注文が一つある
discountedOrder = aCosutomer.discount(anOrder)
                                     // 割引した
```

例 1 では、まず、変数名を見ただけでは役割がわかりません。次に、"0.9" をかける意味がわかりません。前の行に出てくる変数に再度代入していますが、役割が同じなのか異なるのか、わかりません。役割が同じものでも異なるものでも、再代入による副作用が起きないのか疑念が生じます。

一方、例 2 では、最初の行は変数名で役割が「注文」であることがわかります。次の行は変数が「割引した注文」で、「顧客」に「割引」を行った結果であることがわかります。

単純な例ですが、名前で役割がわかるようにすることで、コードを読めば何が起きているのか、何をやろうとしているのかがわかります。

役割をモデリングする

オブジェクト指向でのモデリングは、オブジェクトが役割を演じながらシナリオを遂行することを考えるので、ロール（役割）モデリングということができます。

ロールは通常、オブジェクトの抽象化したものとしてのクラスではなく、関連の役割名称（UML では関連端名）としてモデリングします。

モデリングしていると、ある要素をクラスとロールのどちらにすべきか、迷うことがあります。そのような時は、「そのオブジェクトは生まれた時からそのクラスのインスタンスか？」と考えてみることで判断しやすくなります。「存在が役割に先立つか」どうかで判断するわけです。

例えば、社員 A のオブジェクトを考えてみます。A さんは生まれた時から社員だったわけではありません。会社に所属することになって「社員」になったわけです。つまり、A さんは「社員」という役割を演じていると考えることができます。

社員 A のオブジェクト

A さんは生まれた時から社員だったか？

→ A さんは生まれた時から社員だったわけではない

→ A さんは社員に「なった」

→ 「社員」はクラスではなくロール

アジャイルとモデリング

冒頭で、「アジャイルな開発ではモデリングは不要なのか？」と書きました。

アジャイルな開発とは、暗黙知を形式知にして繰り返しカイゼンしていく開発だと思います。カイゼンは可視化により促進されます。

モデリングは、暗黙知を形式知にするためのツールです。また可視化のツールでもあります。

従って、アジャイルとモデリングは切っても切り離せないもの、と思います。

暗黙知と形式知の交換については、野中郁次郎氏が『知識創造企業』の中で提唱された SECI（セキ）モデルが知られています。アジャイルとモデリングが組み合わさることにより、SECI モデルで示される暗黙知と形式知の交換、および知識移転のプロセスが促進されるものと考えます。

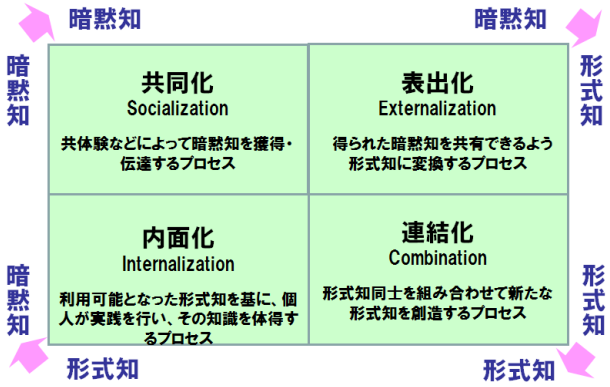

図 1　SECI モデルのイメージ

おわりに

ウォーターフォール、スパイラル、RUP/UP、Scrum といった開発プロセスや開発フレームワーク、SASD、DOA、OOAD といった分析設計手法など、ソフトウエア開発の手法は様々ですが、これらは How の視座であり、同じ対象に対して視野、視点を変えている、と考えることができます。

How に拘る前に、ビジョン、目的、動機など、What、Why の視座、視野、視点を持つことが大切なのではないかと思います。

GSLetterNeo Vol.124
2018 年 11 月 20 日発行
発行者　株式会社 SRA 先端技術研究所

編集者　土屋正人
バックナンバー http://www.sra.co.jp/gsletter
お問い合わせ　gsneo@sra.co.jp

株式会社SRA
〒171-8513 東京都豊島区南池袋 2-32-8

夢を。Yawaraka Innovation
やわらかいのべーしょん