2005－MPS－56（6）
2005／9／21

社会シミュレーションの構築のためのモデルパターン

井庭　崇[††]　津屋　隆之介[†]　青山　希[†]

本論文では，マルチエージェントモデルによる社会シミュレーションの設計ノウハウを,「モデルパターン」として記述することを提唱する．パターンの考え方は，もともと建築デザインのために考案され，その後ソフトウェアデザインに取り入れられたものであるが，本論文ではさらにモデル・デザインへ応用する．モデルパターンを用いることによって，モデル作成者は，設計上の課題に対する解決策を,「新たに見つけ出す」のではなく,「いくつかの候補から選び出す」ことができるようになる．本論文では，実際に 23 個のモデルパターンを提案する．

Model Patterns for Building Social Simulations

Takashi Iba,[††] Ryunosuke Tsuya[†] and Nozomu Aoyama[†]

In this paper, we propose “Model Pattern” for building multi-agent simulations of societies. The concept of “pattern” language was proposed for architectural design, and it had been applied to software design. In this paper, we apply it to model design. With the model patterns, model designers can choose the solution from the model patterns without creating a new way. In this paper, 23 model-patterns are proposed as the instance of model patterns.

1. はじめに

本論文では，マルチエージェントモデルによる社会シミュレーションの設計ノウハウを,「モデルパターン」(model pattern) として記述することを提唱し，実際に 23 個のモデルパターンを提案する．パターンの考え方は，もともと建築デザインのために考案され [1]，その後ソフトウェアデザインに取り入れられたものであるが [2),3)]，本論文ではさらにモデル・デザインへ応用する．

2. モデルパターンの考え方

モデルパターンとは，モデルの「組み立て方」に関する知識をまとめたものである [4)~6)]．そのような知識をパターンとしてまとめる利点は，大きく分けて二つある．一つは，熟練者が自らの経験から得た経験則を明文化しているため，その問題の初心者であっても，効率的かつ洗練された方法でその問題を解決することができるという点である．もう一つは，その設計原理についての共通の語彙を提供するので，これまで直接指し示すことができなかった関係性などについて，簡単に言及することができるようになるという点である．

モデルパターンは，(1) 特徴，(2) 適用条件，(3) 実現方法，(4) 利用に関する注意点・技術的補足，(5) 関連するパターン，の 5 つの要素から構成されている．(1) 特徴には，そのモデルパターンが取り上げる課題と，解決するための利点となる特徴がまとめられている．モデル作成者は，モデルパターンのもつ特徴を比較して，自分の直面する課題に適しているモデルパターンを選択する．(2) 適用条件には，適用条件や適用例がリストアップされている．モデル作成者は，それぞれの条件を閲覧することで，より具体的にモデルパターン同士の比較を行うことができる．(3) 実現方法には，課題に対する解決策の具体的な設計方法が説明されている．(4) 利用に関する注意点・技術的補足には，そのモデルパターンを利用するにあたっての，モデル作成者が行き詰まりやすい点や，気づき難い設計上の注意点を紹介している．モデル作成者は，利用に関する注意点を知ることによって，設計上の陥りやすいミスを回避することができる．(5) 関連するパターンには，よく似たパターンや関連のあるパターンについて紹介している．

モデル作成者は，モデルパターンのカタログから，実現したい部分モデルに関連するモデルパターンを 1 つ以上選び出し，それらを比較して，適すると思うものを採用する．モデルパターンを共有することで，設

† 慶應義塾大学 SFC 研究所
Keio Research Institute at SFC, Keio University
†† 慶應義塾大学 総合政策学部
Faculty of Policy Management

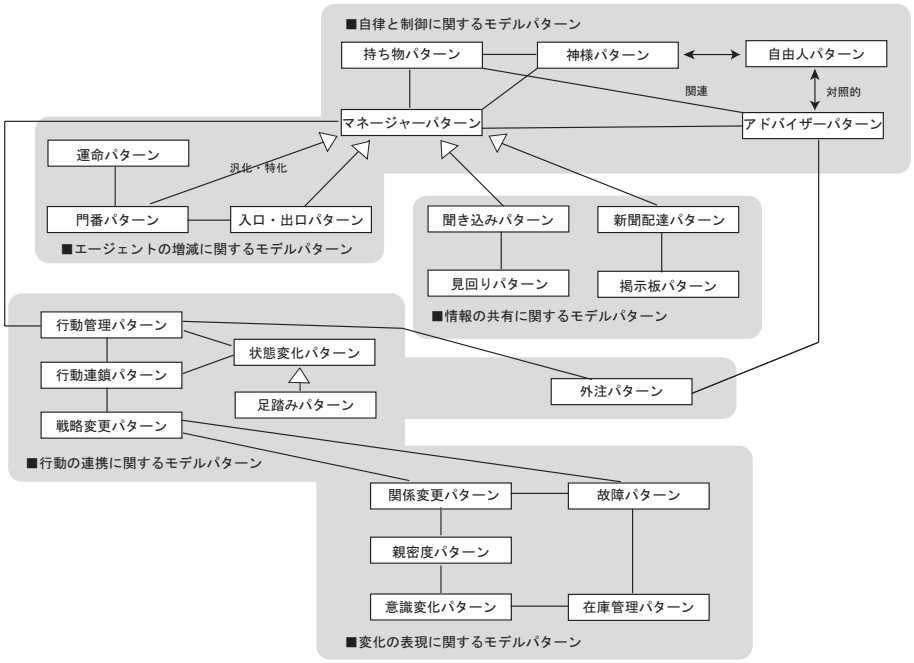

図 1　提案モデルパターン間の全体像と関係性

計上の課題に対する解決策を「新たに見つけ出す」のではなく,「いくつかの候補から選び出す」ことができるようになる.

3. 提案モデルパターン

本論文で提案するモデルパターンは，その設計目的により，いくつかのカテゴリに分類できる．本論文では,「自律と制御」,「エージェントの増減」,「行動の連携」,「情報の共有」,「変化の表現」という 5 つのカテゴリ別に分類して，23 個のモデルパターンを紹介する (**図 1**)☆．以下では，紙面の都合上，モデルパターンの「特徴」と「適用条件」の部分を抜粋することにしたい．

3.1 自律と制御に関するモデルパターン

複数のエージェントが相互作用するモデルにおいて，非対称な関係のモデル化をどのように行うかについて考える必要がある．そのとき，そのような設計のパターンをまとめると，以下のようになる．

3.1.1 神様パターン

主導エージェントが，従属エージェントの振る舞いを直接制御する．個々のエージェント同士の相互作用が複雑であり，一元管理してその計算を行うほうが処理しやすい場合に，このパターンを用いる．また，あらかじめ用意されたデータに沿って，個々のエージェントの属性や振る舞いを操作したい場合などにもこのパターンは適している．

3.1.2 マネージャーパターン

主導エージェントが，従属エージェントの振る舞いを管理し，各エージェントに振る舞いに関する指示を出す．スケジュール表に記載された順番に従ってエージェントの行動を起こさせたり，あらかじめ用意されたデータに一致するように処理を行う場合に，このパターンを用いる．また，エージェント間の相互作用が複雑であり，仲介者を挟んだほうが処理しやすい場合などにも適している．

3.1.3 アドバイザーパターン

管理の役割を担うエージェントが処理を主導するのではなく，個々のエージェントからの問い合わせに反応するというかたちで振る舞いのサポートを行う．個々のエージェントの属性や情報を集計したデータを扱いたい場合に，このパターンを用いる．特に，すべてのエージェントが必ずしもその情報を必要としていない場合に適している．

☆ ここで用いる語彙は，PlatBox 基礎モデル (旧称：Boxed Economy 基礎モデル [7]) に準じている．

3.1.4 自由人パターン

個々のエージェントが自分で自分の振る舞いを管理する．個々のエージェントが，全体の情報を必要とせず，個別に情報を管理して行動するほうがプロセスが扱いやすい場合に用いる．また，個々のエージェントの状態に依存して行動させたい場合に適している．

3.1.5 持ち物パターン

Agent によって管理される要素を，Agent（動的な Entity）ではなく，Goods（静的な Entity）としてモデル化する．シミュレーションを通じて要素の状態が変化しない場合で，それらを種類別に集計量として扱いたい場合に，このパターンを用いる．

3.2 エージェントの増減に関するモデルパターン

シミュレーション実行中にエージェントの数が増減する場合には，それをどのように管理するのかについて考える必要がある．そのとき，そのような設計のパターンをまとめると，以下のようになる．

3.2.1 運命パターン

入場処理と退場処理を一括して管理するエージェントによって，増減の制御を行う．エージェントの状態にかかわらず増減できる場合で，常にエージェント数を一定に保ちたい場合などに，このパターンを用いる．

3.2.2 門番パターン

入場処理と退場処理を一括して管理するエージェントが，要求に応じて入退場に関する指示を行う．入場と退場のタイミングを密接に連携させたい場合に，このパターンを用いる．

3.2.3 入口・出口パターン

入場と退場を異なるエージェントが別々に管理し，その処理が実行されるタイミングのみ設定する．入場と退場の順序を設定し，その間に異なる処理を挟みたい場合に，このパターンを用いる．

3.3 行動の連携に関するモデルパターン

エージェントは，いくつかの行動をもっており，それらを連携させて行動することができる．その際の設計のパターンをまとめると，以下のようになる．

3.3.1 行動管理パターン

エージェントの振る舞いを複数の行動に分割し，それらの実行順序を制御する行動を用意する．エージェントの状態変化によって，行動の順序が動的に変化する場合に，このパターンを用いる．

3.3.2 行動連鎖パターン

エージェントの振る舞いを複数の行動に分割し，それらの連携を個別行動間の連絡で行う．個々の振る舞いの順序の変更が予想される場合や，その間に新しい振る舞いが挿入されることが予想される場合に，このパターンを用いる．

3.3.3 状態変化パターン

振る舞いを Behavior 内のアクションとしてモデル化して，振る舞いの変化を状態遷移として表現する．状態によって振る舞いを変えたい場合に，このパターンを用いる．

3.3.4 足踏みパターン

エージェントの Behavior において，同じ状態で TimeEvent を複数回受け取ることで，ある振る舞いや状態に時間がかかるということを表現する．時間の経過を扱いたい場合や，設定した期間が経つまで同じ振る舞いを続けさせたい場合には，このパターンを用いる．

3.3.5 外注パターン

エージェントの振る舞いを切り出して，他のエージェントにその振る舞いを行わせる．複数のエージェントが同じ機能や資源を同時に利用したい場合に，このパターンを用いる．

3.4 情報の共有に関するモデルパターン

多くの場合，エージェントはほかのエージェントと情報を共有するモデル化が行われる．その際の設計のパターンをまとめると，以下のようになる．

3.4.1 聞き込みパターン

複数のエージェントに送った質問に対する答えを，まとめて受け取る．複数のエージェントに対する質問結果をまとめて受け取り，その結果を集計したい場合に，このパターンを用いる．

3.4.2 見回りパターン

個々のエージェントが，他のエージェントを直接参照することで情報を収集する．他のエージェントの行動を介さずに，そのエージェントの情報を利用したい場合に，このパターンを用いる．特に，シミュレーション実行時の処理を軽くしたい場合に有効である．

3.4.3 新聞配達パターン

情報収集を担うエージェントが情報を収集し，個別のエージェントに結果を知らせる．同一の情報を各エージェントに共有させたい場合に，このパターンを

用いる．

3.4.4 掲示板パターン

情報収集を担うエージェントがあらかじめ情報を収集しておき，個別エージェントが必要に応じて問い合わせて情報を入手する．全てのエージェントが常に情報を必要としているわけではないが，必要に応じて同一の情報を各エージェントが共有したい場合に，このパターンを用いる．

3.5 変化の表現に関するモデルパターン

シミュレーションでは，多くの場合，実行中にいろいろなものが変化する．その際の設計のパターンをまとめると，以下のようになる．

3.5.1 戦略変更パターン

エージェントの取り得る振る舞いを行動単位でモデル化し，それらの行動を切り替えることで振る舞いの変化を表現する．シミュレーション実行時に，エージェントの振る舞いを，あらかじめ用意された行動の候補群のなかから選択したい場合や，今後同じ種類の行動のバリエーションの追加を見込んでいる場合に，このパターンを用いる．

3.5.2 関係変更パターン

エージェント間の関係の変化を，関係の種類の変化として扱う．特定のエージェント間の関係が動的に変わり，それが振る舞いの結果に影響を及ぼす場合に，このパターンを用いる．

3.5.3 故障パターン

Goods の状態の変化を，Goods の種類としてモデル化する．特定の Goods の状態が動的に変わり，それが振る舞いの結果に影響を及ぼす場合に，このパターンを用いる．

3.5.4 意識変化パターン

振る舞いを Information のパラメータとしてモデル化して，振る舞いの変化をパラメータ変化で表現する．振る舞いの変化を，何らかの方程式の定数・変数の変化としてあらわしたい場合に，このパターンを用いる．特に，シミュレーションが進むなかで戦略を生成させたい場合などに用いられる．

3.5.5 親密度パターン

エージェント間の関係の変化を，Information の値の変化として扱う．特定のエージェント間の関係が，定量的な表現で表現され，それが振る舞いの結果に影響を及ぼす場合に，このパターンを用いる．

3.5.6 在庫管理パターン

Goods の状態を，エージェントが記憶する Information としてモデル化する．特定の Goods の状態が動的に変わり，それが振る舞いの結果に影響を及ぼす場合などに，このパターンを用いる．

4. さいごに

本論文で扱っているのは，具体的なモデルそのものではなく，モデル化についての研究である．モデルを用いた研究では，モデルの構築が本質的に重要であるが，モデルを「どのようにつくるのか」(how) についての議論は少なく，いわば「職人芸」となってしまっている．このような問題意識から，私たち，モデルの設計ノウハウの研究に取り組んできた．

今回提案したモデルパターンはどれも，私たちが数多くのモデル構築を行ってきた経験から抽出された知識であり，実際に繰り返し適用されているものである．今後，本論文で提案したモデルパターンが活用されるとともに，このような議論が活発になることを期待したい．

参考文献

1) Alexander, C.: *The Timeless Way of Building*, Oxford University Press (1979). 『時を超えた建設の道』, 鹿島出版会, 1993.
2) Gamma, E., Helm, R., Johnson, R. and Vlissides, J.: *Design Patterns : Elements of Reusable Object-Oriented Software*, Addison-Wesley (1995). 『オブジェクト指向における再利用のための デザインパターン』, 改訂版, ソフトバンクパブリッシング, 1999.
3) Fowler, M.: *Analysis Patterns: Reusable Object Models*, Addison-Wesley (1996). 『アナリシスパターン: 再利用可能なオブジェクトモデル』, 新装版, ピアソンエデュケーション, 2002.
4) 井庭崇: 社会・経済シミュレーションの基盤構築：複雑系と進化の理論に向けて, 博士論文, 慶應義塾大学 政策・メディア研究科 (2003).
5) 岡部明子, 井庭崇: 社会・経済シミュレーションのモデル・パターン：複雑系における動的な変化を記述する, 第 2 回情報科学技術フォーラム Forum on Information Technology (FIT2003) (2002).
6) 津屋隆之介: 社会シミュレーションの作成を支援するタイプとパターン, 修士論文, 慶應義塾大学政策・メディア研究科 (2005).
7) 井庭崇, 中鉢欣秀, 松澤芳昭, 海保研, 武藤佳恭: Boxed Economy Foundation Model: 社会・経済のエージェントベースモデリングのためのフレームワーク, 情報処理学会論文誌: 数理モデル化と応用, Vol. 44, No. SIG14(TOM9) (2003).