修士論文概要

概念モデリングにおけるソフトウェアパターンの再利用を支援するツール†

赤木　謙*

(2020年３月２日受理)

Preliminary Evaluation of a Support Tool foe Reuse of Software Patterns in Conceptual Modeling

Ken Akagi

(Received March 2, 2020)

1　はじめに

1.1　背景

ソフトウェアパターン（以下、パターンと呼ぶ）とは、開発過程において頻繁に表れる構造を抽出したものである。パターンを利用することで、再利用性の高い設計を行うことができる。しかし、設計の段階で既存のクラス図にパターンを適用するためには、ほかのクラスとの相互関係や依存関係を考慮しながら、パターンの具体化作業を行わなければならない。本稿で述べる具体化作業とは、パターンで定義されている抽象的なクラス構造をソフトウェアとして動作可能なレベルにまで具象化する作業のことである。具体的には、パターンのモデルに具体的なクラス名を付けたり、モデルに機能の実現に必要なメソッドを追加する等の作業が該当する。もし、既存のモデルに対して追加でパターンの導入を行なう場合、既存のモデルに対しても変更を加える必要が出てくる。加えて、パターンには変更しても良い部分とそうではない部分が混在しており、設計者はこれらの制約を考慮しながら作業を進める必要が出てくる。従って、開発者がこのような作業を行うことはとても困難である。

1.2　研究目的

前述の問題を解決するための一つの方法として、パターンを支援するモデリング環境を導入することが考えられる。このようなツールは、これまでにも様々なものが提案されているが、多くのツールはあらかじめ保存しておいたパターンを呼び出す機能に留まっており、前述の具体化作業は設計者が行う必要があった。加えて、前述のとおり設計者が変更してはいけない部分を変更してしまうなどの操作によって、パターン本来のメリットが得られない問題が発生する可能性もあった。

そこで、本研究は、既存のクラス図にパターンを適用させるよう再構成を行うモデリング環境を実現する。そして、実装したツールの有効性を評価することで、本手法の効果を検証する。

2　提案手法

2.1　パターンのスポットと現状の問題点

本研究では、パターンに含まれる変更しても良い部分とそうではない部分をスポットとして区別し、前者をホットスポット、後者をフローズンスポットと呼ぶ。

つまり、ホットスポットはパターンの適用の際には必ず変更すべき場所であることを表しており、逆にフローズンスポットはいかなる場合であっても変更すべき場所でないことを表している。

このようなパターンのスポットに着目し、クラス図上でパターンの利用を支援する研究として、讃井らの研究[5)]が挙げる。讃井らの手法では、パターンの具体化操作をあらかじめ計算機上に蓄積させ、設計時にパターンを扱う際に具体化操作を呼び出すことで、パターンのホットスポットの具体化を自動で行い、パターンの利用を支援するものである。ただし、パターンの具体化操作はあらかじめ計算機に登録しておかなければならず、登録されていない具体化操作を行いたい場合には、改めて登録する必要があった。また、既存のモデルにパターンを後から適用する際には、パターンをエディタ上に呼び出して具体化を行なった後に、既存のモデルと合成するという２段階の作業を経る必要があった。

2.2　本研究が提案する手法

本研究では、パターンそれぞれ持っている固有のスポットに着目し、パターンのモデルを呼び出すときに、既存のクラス図の中でパターンを適用したい箇所と、パターンの変更可能なスポットを自動で合成を行う仕組みを提案する。

† 本研究の一部は以下において発表した
・The 2020 RISP International Workshop on Nonlinear Circuits, Communications and Signal Processing (NCSP 2020)

* 電子情報メディア工学専攻 2188002 橋浦研究室

具体的には、パターンのホットスポットを「クラスA」、既存のクラス図の中でパターンの部分を構成するクラスを「クラスB」とする。本手法ではクラスAをクラスBに置き換え、クラスAの被関連先をクラスBに変更することで、既存のクラス図にパターンを適用することが可能になる。また、パターンのホットスポットであるクラスの中に、フローズンスポットとなるメソッドや属性が存在する場合がある。そのため、クラスAにメソッド、或いは属性が存在していたとき、合成した後のクラスBにメソッド、或いは属性を追加する。それから、パターンのホットスポットが複数箇所存在する場合、それぞれのパターンの部分を構成するクラスを選択することで、対応づけることで合成可能とする。

一例としてObserverパターン[3]を適用する場合の例を図１に示す。Observerパターンを利用するクラスはConcreteSubjectクラスとConcreteObserverクラスであり、既存のクラス図の中でパターンの部分を構成するクラスはParticipationDataクラス、ItemEvalFrameクラス、SeminarEvalFrameクラスである。したがって、ConcreteSubjectクラスをParticipationDataクラスに、ConcreteObserverクラスをそれぞれItemEvalFrameクラス、SeminarEvalFrameクラスに置き換え、それぞれのクラスの被関連先を変更することで、既存のクラス図にObserverパターンを適用することが可能である。

３　実装手法

本手法を実装したツールを実装した[1)2)]。実装形式はWebアプリケーションとし、実装のベースとなるUMLエディタとして田中ら[4]が開発したKIfUを使用した。KIfUとは、概念モデリングの編集過程のデータを細粒度に収集し、編集過程を明らかにすることを実現するために開発された。クラスの作成、属性・メソッドの追加、関連の作成など、UMLエディタに必要な機能がそろっている。本ツールはKIfUに、以下の三つの機能を実装した。

１．データベースにモデルを登録する機能
２．モデルと合成個所の選択を行う機能
３．合成を行う機能

本ツールを使う流れについて述べる。まず、登録されているフレームワークのモデルの中から適用させるモデルを選択する。次に、選択したモデルのホットスポットと対応するクラスの選択を行う。最後に合成ボタンを押すことで、選択したモデルのホットスポットが対応付けしたクラスに置換される。

４　実験と評価

本手法の有効性を検証するために、提案手法を実装したKIfUと従来のKIfUでの比較実験を２回行った。１回目の実験ではデザインパターンのモデルを扱う実験（実験１）を行い、２回目の実験ではフレームワークを適用する実験（実験２）を行った。被験者はどちらも日本工業大学の情報工学科の学生である。これらの被験者はオブジェクト指向についての多少の知識はあるものの、パターンについて特段の知識を持たない者とした。

実験の目的は、２つ存在する。１つ目は、パターンを利

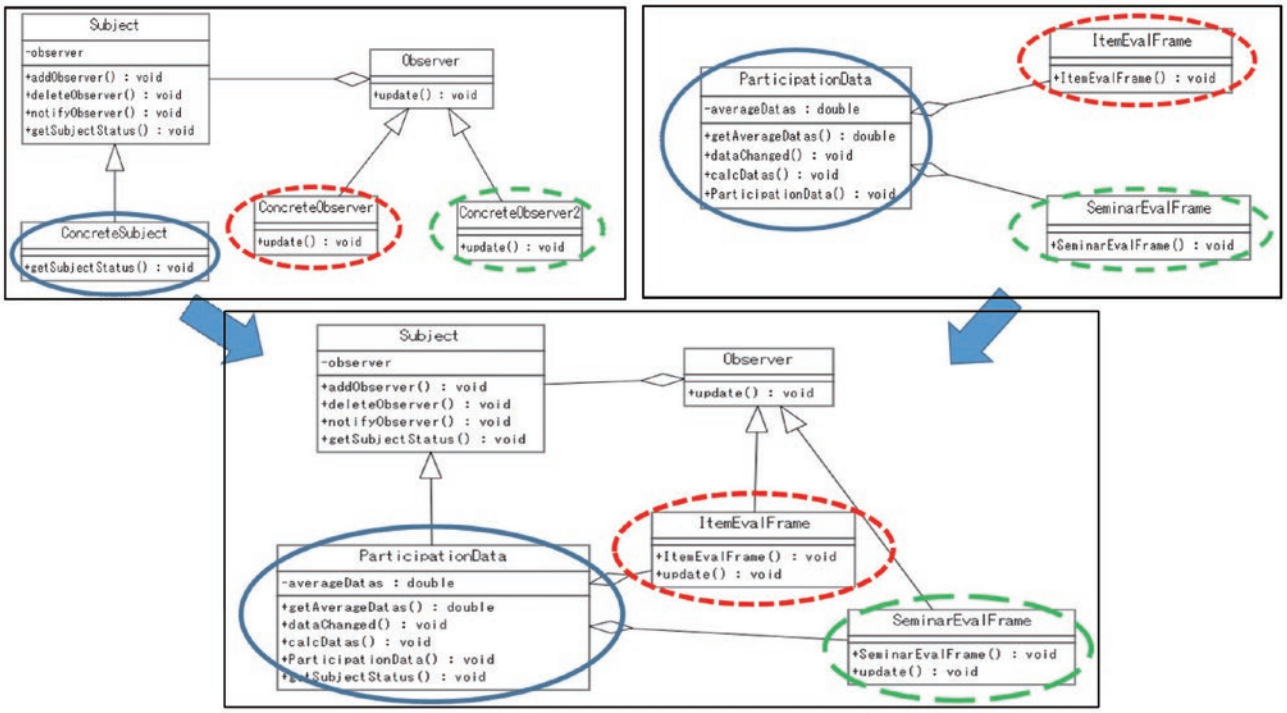

図１　Observerパターンを適用する例

用した設計において、本ツールが従来のUMLエディタよりも高い効果を示すことを確認すること。２つ目は、パターンの知識がない被験者にツールを使用させることで、パターンに不慣れな者であってもツールの効果が得られるかどうかを確認することである。

実験１の流れは以下の通りである。まず、被験者を本提案手法を使用しないグループ（グループA）と、使用するグループ（グループB）に分け、それぞれのグループに対して、あるモデルに対して、あるパターンを適用するものである。演習問題は２問とし、問題１はクラス図の要素とパターンのモデルからクラス図を作成してもらう問題、問題２は既存のクラス図にパターンを適用させる問題を行ってもらった。

実験２の流れは以下の通りである。まず、被験者を２つのグループ（グループAとグループB）に分ける。そして、それぞれのグループに対してフレームワークを適用する演習を行ってもらう。演習問題は２問とし、問題１はAWTというフレームワークを利用した電卓プログラムを設計してもらう問題、問題２はEclipseのプラグイン開発の設計してもらう問題を行ってもらった。一回目は従来のKIfUで設計を行い、二回目では合成機能を実装した本ツールで設計を行ってもらった。

実験１、２どちらも実験後にそれぞれの成果物のデータを収集し、それらの差を比較する。

４.１　仮説

ここで、本ツールを使うことで期待される効果を述べる。今回の実験では、本ツールの期待される効果として仮説を２つ立てた。１つ目の仮説（仮説１）は、モデルを利用して設計を行うことで、モデルの誤りが少なくなることである。これは、パターンのモデルを設計者自らが作成して設計を行うよりも、あらかじめ登録しておいたパターンをモデルに自答的に適用した方が、設計者はパターンに関する制約を考慮する必要がなくなるため、誤りが少なくなるというものである。本研究におけるモデルにおける正確性については、次節で述べる。２つ目の仮説(仮説２)は、ツールは設計を行う時間の短縮になることである。本ツールは、既存のクラス図にパターンのモデルを適用するプロセスを自動化している。そのため、パターンを利用した設計においては、従来のUMLエディタより作業を簡略化して設計を行えるため、設計の時間の短縮につながるのではないかと考えられる。

４.２　評価基準

ここでは、本研究で扱う成果物の正確性を評価するための評価基準について述べる。パターンの適用については、適用される既存のモデルによって具象化されたパターンの形は異なるため、前述のスポットなどに関する制約を考慮した評価基準を作成した。今回の利用した評価基準の観点は大きく「合成」、「フローズンスポット」、「ホットスポット」、「関連」の４つとし、その具体的な基準については表１に示した。

４.３　実験結果

各被験者の平均時間を測定した結果を表２に示す。表から、実験にかかった時間はどちらの実験でもグループ間に差はなかった。

表２　平均時間

	従来のKIfU	本ツール
実験１	33：26	27：41
実験２	32：19	49：38

それぞれの実験の評価基準ごとの成果物の評価の平均とその差を表２、表３に示す。表から実験１ではホットスポットと関連に、実験２では合成とフローズンスポットに有意差があることが確認できた。

表３　実験１での成果物の評価

	グループA	グループB	差（A-B）
HS	3.125	3.8	-0.675（**）
関連	2.125	3.3	-1.175（**）
			** α < 0.05

表４　実験２での成果物の評価

	一回目（A）	二回目（B）	差（A-B）
合成	3	3.9091	-0.9091（**）
FS	3.2727	4	-0.7273（**）
			** α < 0.05

５　考察

仮説１では、実験結果から全体的にツールを使ったときのほうが評価が高いことがわかる。また、実験結果から実験１ではホットスポットと関連に、実験２では合成とフローズンスポットに有意差があることがわかった。このこ

表１　評価基準

レベル	合成	フローズンスポット	ホットスポット	関連
1	モデルの合成が行われていない	フローズンスポットがすべて存在しない	ホットスポットがすべて存在しない	関連が引かれていない
2	モデルをそのままの状態にしている	フローズンスポットが一部存在しない	ホットスポットが一部存在しない	クラス間にひかれていない関連が存在する
3	合成箇所が模範解答と一致していない	クラス名・メソッドの名前を変更している	クラス名を変更していない	模範解答とは異なるクラス間で関連が引かれている
4	合成箇所が模範解答と一致している	変更していない	問題にあった名前に変更している	模範解答と一致している

とから、仮説１は正しいことが示唆された。

実験１でホットスポットと関連に有意差が現れた要因として、本ツールを利用することで、モデル内のホットスポットが既存のクラスの要素に置き換わり、モデルそのものを記述することにより関連に対して余計な変更が生まれなかったためではないかと考えられる。

実験２で合成とフローズンスポットに有意差が現れた要因として、本ツールを利用しモデルを提供することで、クラス間の依存関係の把握が用意になったことや、フローズンスポットの場所が明確になったためだと考えられる。

また、実験１と実験２で有意差に関して逆の結果を示していることがわかった。このような結果となった要因として抽象度の違いが考えられる。フレームワークはデザインパターンの要素を組み込んでいるため、フレームワークのほうがデザインパターンより抽象度が低い。抽象度が高いとモデルの構造がより単純になるため、具体化手順が煩雑になる。一方で、抽象度が低いとモデルが具体的になるため、具体化しなければならない箇所が多くなる。これらを本ツールによって支援できたことで、以上のような結果が得られたのではないかと考えた。

仮説２では、どちらの実験でも被験者の平均時間がグループ間で差が見られなかった。実験２では、本ツールを使ったほうが時間がかかっていることから、本ツールは時間の削減にはならないという事がわかった。従って、仮説２は立証することができない。

５.１　関連研究

関連研究として山本らの研究[6)]を挙げる。この研究では、登録済みのデザインパターンのクラス図と設計者が作成したクラス図との対応関係を入力していくことによってパターン適用を支援している。本研究と比較すると、ホットスポットのみを設計者が容易にカスタマイズできるようにしている点が類似しているが、山本らの手法では、ホットスポットの属性やメソッドはツールを使ったあとに、設計者自身がまたカスタマイズしないといけない問題がある。また、最初からパターンを組み込むときのみを想定しているため、一部分にパターンを埋め込むような操作ができない。本手法では、既存のクラス図にパターンを跡から適用でき、既存のクラスの属性やメソッドを合成後のクラスに追加するようにしている。

６　おわりに

本研究では、既存のクラス図に後からパターンを適用させる手法を提案し、従来のUMLエディタであるKIfUを拡張して提案手法を実装したモデリング環境を実現した。そして、本ツールの有効性を検証するため、実験を実施し、評価を行った。その結果、本ツールは設計を行う時間には影響を与えないが、設計図の誤りが少なくなるいことを確認した。このことから、設計の支援ができているのではないかと結論づけた。今後の課題としては、本ツールのマニュアル機能を組み込むと共に、他の様々なパターンにおいて本手法が有効かどうか検証していきたい。

謝　　辞

本研究の一部はJSPS科研費JP18K11579助成を受けた。

参　考　文　献

1）赤木謙，橋浦弘明，田中昂文，櫨山淳雄，高瀬浩史，“ソフトウェアパターンの再利用を支援するモデリングツール，”2018年電子情報通信学会総合大会論文集，p. 115, 2018-3.
2）赤木謙，橋浦弘明，田中昂文，櫨山淳雄，高瀬浩史，“概念モデリングにおけるソフトウェアパターンの再利用を支援するツールの初期評価，”信学技報，vol. 118, no. 471, pp. 55-60, 2019-3.
3）Gamma, E., Helm, R., Johnson, R., and Vissides, J.：Design Pattern：Elements of Reuable Object-Oriented Software, Addison-esley, 1995.
4）Takafumi Tanaka, Hiroaki Hashiura, Atsuo Hazeyama, Seiichi Komiya, “Do Learners to Create an Artifact with Good Quality Make a Number of Trials and Errors during the Editing Process?,” 2015 3rd International Conference on Applied Computing and Information Technology/2nd International Conference on Computational Science and Intelligence, pp. 28-33, Nov. 2015.
5）讃井崇喜，加茂昌彦，小林隆志，佐伯元司，“デザインパターンのモデル化と適用支援ツール，”電子情報通信学会技術研究報告．SS，ソフトウェアサイエンス，pp.1-8, 2000-5.
6）山本純一，松本一教，“CASEツールによるデザインパターン適用支援，”情報処理学会研究報告ソフトウェア工学，vol1996, no84, pp.41-48, 1996-6.

論文審査委員

審査委員（主査）　准教授　橋浦　弘明
審査委員（副査）　教　授　高瀬　浩史
審査委員（副査）　教　授　粂野　文洋