B-21　プログラミング段階でのデザインパターン適用支援

Aiding Design Patterns in Programming

山下 純司*　谷川 健†　高木 俊幸‡　林 雄二†
Junji Yamashita　Takeshi Tanigawa　Toshiyuki Takagi　Yuji Hayashi

1. はじめに

プログラミング段階におけるデザインパターン適用支援ツールの開発をおこなったので報告する。

デザインパターン[1][2]は良質のオブジェクト指向設計を，テンプレートに従い記述しカタログ化することで再利用を可能としたものである．しかし，あらゆる状況で最も適切なパターンを選択するには，多くのパターンを熟知しておく必要がある．すなわちオブジェクト指向技術の有効活用法として，デザインパターンを利用することはそれほど簡単ではない[3].

このようなことから，デザインパターンに対する適用支援のツールが期待されている．国内のデザインパターンの適用支援の先例に小林[4]や永山[5]等の研究がある．これらはオブジェクト指向分析・設計のモデリング段階での適用支援であり，デザインパターンを熟知していないプログラマやオブジェクト指向プログラミングの初学者が利用することができるツールとはいえない．また，前記のデザインパターンの適用を支援するツールの開発・研究のほとんどは分析・設計の段階で利用するものである．しかし現実のソフトウェア開発では，最初から完全な設計を行うよりもプログラミングを通して設計を洗練することが行われる．また，オブジェクト指向プログラミングを学ぶ段階で，良い設計例としてのデザインパターンを知ることは有用であろう．そこで筆者らは，プログラミング（Java 言語）段階でデザインパターンの利用を補助するためのツールを開発した[6].

本ツールは，プログラム開発段階での設計に対する要求が生まれたときに，その要求に対応するデザインパターンを提案し，パターン適用の結果のプログラムの構造を元のソースプログラムとコメント文によって提示することを可能にしたものである．デザインパターン適用後の Java ソースプログラムの構成は，独自に開発したデザインパターン記述用テンプレート記法で表現されている．

2. 支援ツールの概要

本ツールの構造を図１に示す．本ツールは，まずユーザから，Java のソースファイルとデザインパターン毎に定義される要求項目及びパラメータを受け取る．そして Java のソースから構文木を生成し，要求項目からパターンテンプレートを選択する．デザインパターン適用クラスが，これらの構文木，パターンテンプレートとパラメータからデザインパターンの適用例を出力する．Java ソースを解析し構文木を生成するパーサは JavaCC[7] を用いて作成している．

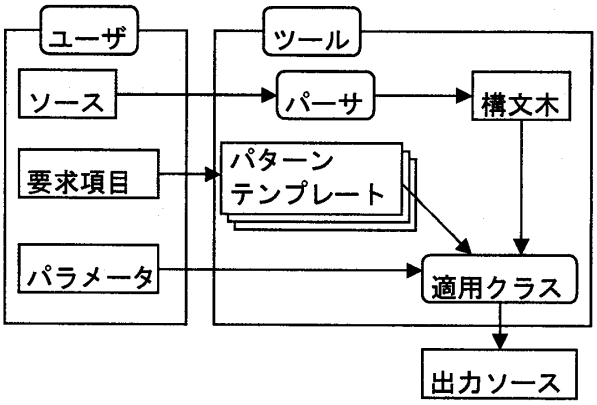

図１ 支援ツールの構成

本ツールでは GoF[1] のデザインパターンを対象としている．GoF のデザインパターンは，生成・構造・振舞に分類されており，本ツールでもこの分類を利用している．ユーザにはソースファイルへの変更要求の項目をメニュー形式で提示する．この要求項目はデザインパターンの適用可能性を元にしており，例えば以下のような項目を用意している．

- 生成
  1. インスタンスの生成を一般化しておきたい．(Abstract Factory)
  2. 同じ状態を持つ複数のインスタンスを作りたい．(Prototype)
- 構造
  1. 2 つのクラスのインターフェイスが異なるので共通にしたい．(Adapter)
  2. 実行時に機能を追加・変更できるようにしたい．(Decorator)
- 振舞
  1. メソッドを他のメソッドと切替えが可能になるようにしたい．(Strategy)
  2. 同じ条件判定による分岐が各メソッドに存在しているので簡素化したい．(State)

本ツールでは，まずメニューから分類を選ぶと個別の要求項目(パターン)の一覧を提示しそこから項目を一つ選択してもらう(図２)．項目の選択により適用するパターンテンプレートが決まるので，パターンテンプレートを適用するために必要なパラメータを問い合わせる(図３)．

3. 使用例

いまクラス Rule に，三目並べの盤面が与えられると勝敗を判定する judge() というメソッドがある．judge() メソッドの勝敗判定アルゴリズムを，より高速なアルゴリズムの実装実験などのために，適時切替えることができるよう

* ㈱エヌアイエス
† 北海道情報大学経営情報学部
‡ 北海道情報大学大学院経営情報学研究科

にしたい．この例を用いて本ツールの使用法と結果を説明する．

まず本ツールでは，アルゴリズムの切替えは振舞にかかわることなので，メニューから「要求項目」の「振舞」を選択する．すると具体的な項目のリストが表示される(図２)．この中から，この例ではメソッドの実装を切替えたいので，「○メソッドを他のメソッドと切替えが可能になるようにしたい(Strategy)」を選択する．ここで本ツールはパターン適用に必要なパラメータとして切替えを可能にしたいメソッド名とそれがあるクラス名の入力をユーザに促す(図３)．今，勝敗判定のアルゴリズムを担っているのは Rule#judge() であり，実装を切り替えたい部分なので，それぞれ judge と Rule が入力されている．

以上の入力から，ツールは 図４ のように Strategy パターンの適用例を出力する．適用例にはアルゴリズムをクラス化するための新たに作成されたクラス Judge と，その中に Rule#judge() の実装を引き継ぐ Judge#do() のスケルトンコードが含まれている．その他，クラス Rule の修正例をコメントの形で付加している．クラス名 Judge はツールに入力されたパラメータ(メソッド名 judge)から作られた．メソッドの中身を移動させやすいように Judge#do() は Rule#judge() と同じパラメータリストになっている．

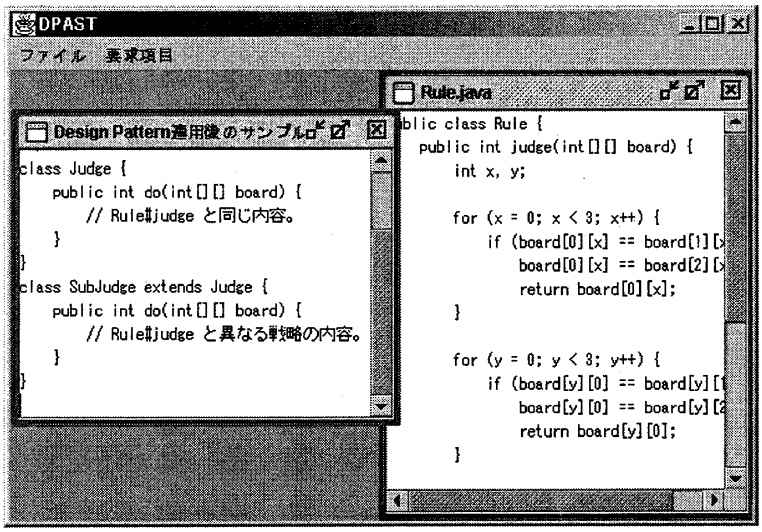

図４ 適用例の出力

4. おわりに

本稿では，プログラミングの局面で，設計を考慮したプログラム構造の変更を行う場合に，デザインパターン適用のヒントをプログラマに提供することを意図したツールの開発を報告した．すなわち，パターンの意味からその利用方法を考えるのではなく，本ツールは，設計の必要性から適当なパターンを探し出すことができるものである．現在，約 15 種類のパターンに対応している．

デザインパターンを適用してプログラムの構成を変更する場合にはリファクタリング[8]の適用が有効である．また，細部に渡る影響を含むプログラムの変更を自動化出来ることが望ましい．これはリファクタリングの適用ツール開発という課題であり，必ずしも容易なことではない．従って，現段階では変更された形のプログラムをコメント付きのプログラムとして生成している．

開発の目的はパターンの適用支援であり，ユーザインターフェイスとパターンの適用を実行する部分を作成した．しかし，独自のユーザインターフェイスを提供することはあまり好ましくない．なぜなら，パターンの適用支援の対象となる利用者はおそらく Java の統合開発環境(IDE)，あるいは Emacs のようなテキストエディタを利用して開発しているからである．既存のツールとの親和性を高めていくことは検討課題の一つである．

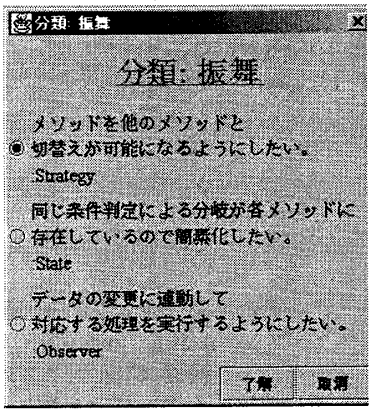

図２ 要求項目の選択

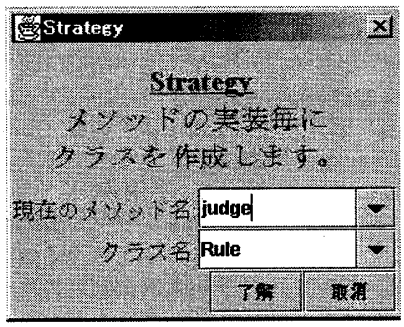

図３ パラメータの入力

参考文献

[1] E.Gamma,R.Helm,R.Johnson,J.Vissides:Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley(1995)

[2] F.Buschmann, R.Meunier, H.Rohnert, P.Sommerlad, M.Stal: Pattern-Oriented Software Architecture, John Wiley & Sons(1996)

[3] 中村宏明,佐藤啓太,高木浩光,友野晶夫,細谷竜一:パターンは役に立ったのか？,オブジェクト指向 2001 シンポジウム資料集,pp.141-156(2001)

[4] 小林 隆志, 讃井 崇喜, 佐伯 元司:デザインパターン適用支援ツール,オブジェクト指向シンポジウム 2000 資料集, pp.183-184 (2000).

[5] 永山 英嗣, 原田 実:デザインパターン適用における設計図融合と最適パターン探索の支援系 OOPAS, オブジェクト指向シンポジウム 2000 論文集, pp.157-164 (2000).

[6] 山下 純司, 林 雄二:デザインパターン適用支援ツールの開発,情報処理北海道シンポジウム 2001 講演論文集, pp.107-108 (2001).

[7] http://www.webgain.com/products/java_cc/

[8] M.Fowler: Refactoring: Improving The Design of Existing Code, Addison Wesley(1999)