

Title	パターンを利用したクラスブラウザの設計と実装
Author(s)	萩原, 豊隆
Citation	
Issue Date	1997-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/1007
Rights	
Description	Supervisor:篠田 陽一, 情報科学研究科, 修士

パターンを利用したクラスブラウザの設計と実装

萩原 豊隆

北陸先端科学技術大学院大学 情報科学研究科

1997年2月14日

キーワード: デザインパターン, オブジェクト指向, リバースエンジニアリング.

1 研究の背景と目的

近年、オブジェクト指向技術の適用例が多くなった。ソフトウェア技術者は、オブジェクト指向技術を、ソフトウェア開発のさまざまな段階で積極的に利用している。特に、大規模なソフトウェアの実装で、オブジェクト指向プログラミング言語を利用している。

オブジェクト指向プログラミング言語を使用した実装では、クラスライブラリが重要な役割を果す。また、過去に開発したソフトウェアに含まれているクラス群も重要な要素である。これらのクラス群は、新規にソフトウェアを開発するときに、再利用の対象となる。しかし、クラス群を再利用するためには、プログラマがそのクラス群がどのような設計をもつかを理解しなければならない。通常、設計者は高度な機能を実現するため、協調して振る舞うクラス群を設計する。もし、このようなクラス群の設計を理解できれば、個々のクラスの機能を使用するだけでなく、協調して振る舞うクラス群としての複合的機能を使用できる。そこで本研究では、プログラマが、協調して振る舞うクラス群を理解するため支援法を提案する。

2 関連研究

デザインパターンは、協調して振る舞うクラス群の設計に利用する。この知識は、既存のソフトウェアに含まれるクラス群をプログラマが理解する時に役立つ。また、オブジェクト指向パターン Party は、協調して振る舞うクラス群の計算機を用いた抽出処理に有用な概念である。

3 支援法の検討

研究の対象とするプログラミング言語には Java を採用した。これは以下の理由による。Java は強い型付けをもつ言語であり、ソースコードから情報を収集しやすく、既存ソフトウェアに含まれるクラス群からの情報収集に有利である。また、すべての要素をクラスとして扱う。そのため、C++ の様な複合的プログラム言語と比較して、クラス群に対する支援機構構築に専念しやすい。

デザインパターンでは、以下に示す 2 つの設計原理がある。第 1 点として、オブジェクトのクラスとオブジェクトの型を区別して考えること、第 2 点として、オブジェクトの型に対して、オブジェクトコンポジションを使用することである。Java は、インターフェース機構により、クラスと型を分離して扱うことができる。オブジェクト指向パターン Party は、この分離機構を考慮していないため Party を拡張した。これにより、Party でクラスと型の区別できるようになった。また、Party の連鎖をたどることで、プログラマはオブジェクトコンポジションに関する情報を取得できる。このことは、デザインパターンに関する知識の利用促進に有効である。

4 クラスブラウザの設計

プログラマへの支援機構として、クラスブラウザを設計し、実装した。このクラスブラウザでは、Party に基づいたブラウジングが可能である。Party の連鎖をたどることによって、協調して振る舞うクラス群を表現できる。

Java のインターフェースに対応した Party の拡張性を検討するために、クラス群に対し、試作したクラスブラウザを適用した。その結果、インターフェースを利用している部分が容易に見えてきた。さらに、インターフェース周辺の Party の連鎖をブラウジングすることで、デザインパターンで頻繁に使用される構造を認識できた。しかし、実際に使用されているデザインパターンの特定には、クラス名を参考にしたり、実際にコードを参照する必要があった。

5 クラス群への適用と評価

また、既存のクラスブラウザと比較し、試作ブラウザの評価を行った。その結果、試作ブラウザは既存のクラスブラウザに比べ、協調して振る舞うクラス群の理解に対する支援能力に優れるとの結論を得た。既存のクラスブラウザには、関数呼出しの連鎖をグラフ表示する。しかし、そのグラフは、協調して振る舞うクラス群の一部を表示するにすぎない。ため、クラス群の構造を理解するには不十分である。

6 結論と今後の課題

クラスブラウザ実装によって以下の結論を得た。

- クラスブラウザへのオブジェクト指向パターン Party の連鎖を導入した。その結果、協調して振る舞うクラス群に関して、理解に必要な情報をプログラマが得ることができた。
- Java 言語のインターフェース機構を Party へ導入した。導入により設計者の目的、動機を推測するための補助が可能となった。
- クラスブラウザによる Party の表示は、関数の呼出しグラフ表示に比較して、協調して振る舞うクラス群の理解を支援するという点で有効である。

今後の課題として、以下の課題を挙げることができる。

- Java 以外の言語で利用するには、オブジェクトのクラスとオブジェクトの型の区別の表現が可能なように、Party の拡張方法を検討する必要がある。
- デザインパターンの設計原理の一つである「オブジェクトコンポジションによる要求の委譲」を適切に扱う手段を考案する必要がある。