金融アプリケーション開発におけるパターン適用

上原 忠弘* 山本 里枝子* 赤木 豊** 石田 修一*** 藤田 雅人*** 吉田 裕之*
6AE－1 (株)富士通研究所*, (株)富士通金融システムズ**, 富士通(株)***

1.はじめに

パターンはある状況下で繰り返し起こる問題とそれに対する解法の組であり，近年オブジェクト指向開発の再利用技術として注目されている．本稿では，複数のパターンを戦略的に用いることにより，効率的にアプリケーション開発を行った事例を報告する．

アプリケーション開発は一般に各工程で前工程での成果物を詳細していく作業である．その作業を効率化するためにパターンを部品として用いる．一度検討した分析・設計事項をプロジェクト内のパターンとして抽出し，同じプロジェクト内で同じような状況に出会った場合にはそのパターンを適用することで開発効率を上げる[1]．

我々はこれに加えて，パターンを工程に従って戦略的に順次適用していく方式を検討した．

ここで適用されるパターンは大きく分けて以下の種類に分類される．

分析パターン： ある問題領域でアプリケーションの共通な構造や振る舞いを規定するパターン．

設計パターン： アプリケーションのある一部分の検討項目について，より詳細な構造や振る舞いを規定するパターン．

実装パターン： 特定の開発言語で，アプリケーションの一部分の構造や振る舞いの表現方法を規定するパターン．

前工程の結果に対してパターンを適用しその工程の成果物を得ることを繰り返してアプリケーションを詳細化していく．

各レベルのパターンを用意して適用の順序をあらかじめ規定しておけば，アプリケーションの詳細化を非常に効率的に行うことができる．

この場合，そのような体系化されたパターン群を抽出，構成することが問題になる．一般的に適用できる洗練されたパターンを生成することは非常に難しい．しかし特定の文脈に絞ればパターンを生成しやすくなる．あるパターンを適用することを前提とすれば文脈が明らかになり，その結果に適用するパターンを生成しやすくなる．

2.金融アプリケーションへの適用

我々は，金融アプリケーション開発にオブジェクト指向技術を適用した．そのなかで上記の観点でパターンを抽出，適用した．

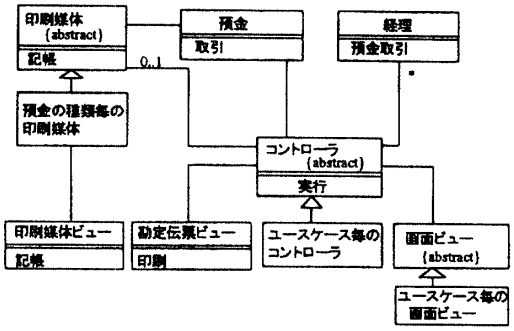
2.1.開発方法の概要

まず分析では，ユースケースを用いて要求を分析し，各ユースケースごとにオブジェクトを抽出し，オブジェクト間の協調を検討した．設計ではプラットフォーム，DBMSを考慮して分析結果を詳細化した．実装では，OOCOBOLを用いてコーディングを行った．

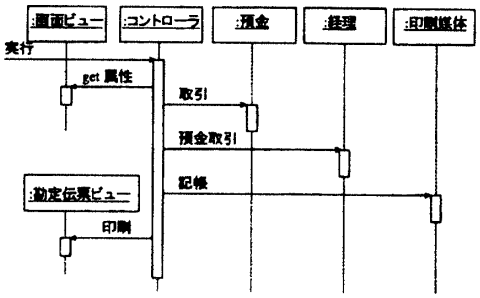
2.2.預金業務取引パターン（分析パターン）

預金業務取引パターンは，システム化範囲内の殆んど全てのユースケースに適用した．一ユースケースは一取引("普通預金業務の入金"など）に対応し，取引毎に画面操作が異なる．また預金種類により印刷物の仕様が異なる．

第1図にパターンの構造の概要を示すクラス図を，第2図に協調の概要を示すシーケンス図を示す．なお表記法はUML1.0を用いる．



第1図預金業務取引パターン



第2図 預金業務取引パターンの協調

第1図で，コントローラクラスはそのユースケースの処理を制御するクラスであり，ユースケース間の共通の振舞いを抽象クラスとして定義している． 各ユースケース固有の処理はこれを継承する具象クラスを定義して実現する．ビュー ク

Using patterns in a financial application development
Tadahiro Uehara, Rieko Yamamoto, Hiroyuki Yoshida, Fujitsu Laboratories Ltd.
Yutaka Akagi, Fujitsu Financial Systems Ltd.
Shuichi Ishida, Masando Fujita, Fujitsu Ltd.

ラスは画面や印刷物の仕様の管理と入出力を制御するクラスであり，ユースケース間で共通なもの（勘定伝票ビュー，印刷媒体ビュー），異なるもの（画面ビュー）がある．

第２図では，まずユースケースのアクター(テラー)からのメッセージをコントローラが受ける．コントローラは画面から情報を受けとり(get 属性)，さらに他オブジェクトにメッセージを送信して，一トランザクション中での各オブジェクトの振舞いを制御する．

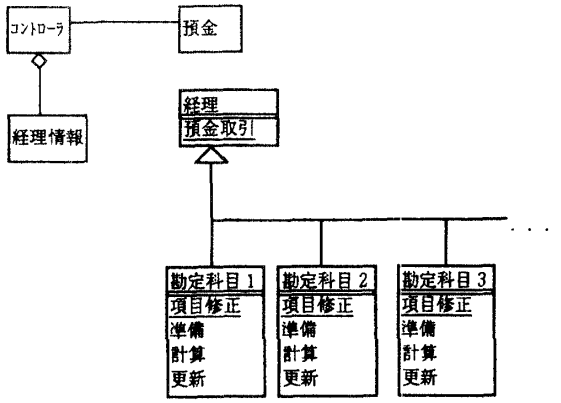
預金業務取引パターンは，複数のユースケース間に共通のアーキテクチャを提供した．これにより，各ユースケース毎に異なる実装となりがちなオブジェクト構造と振舞いが統一された．

2.3.経理パターン(設計パターン)

この節では今回適用された設計パターンの１つを取り上げる．

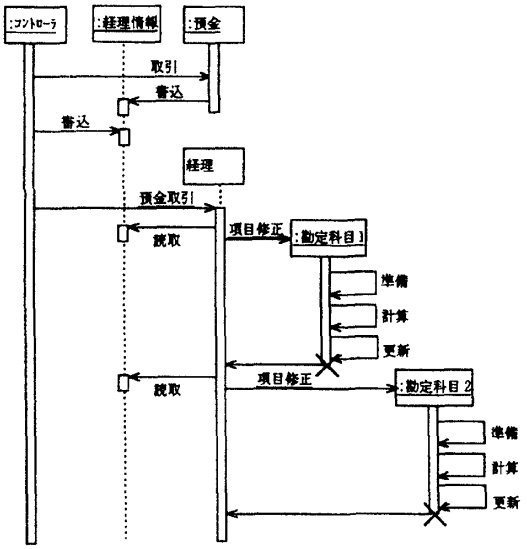
分析パターンはアプリケーションの振る舞いの概要を規定したものであり，抽象度が高い．アーキテクチャをさらに詳細化するために，その上に設計パターンを適用する．例えば，預金業務取引パターンでは預金が処理を行ったあと経理が金銭の異動の記録を取る順序が規定されている．経理パターンは経理クラスの振る舞いを詳細化し，勘定科目と呼ばれる項目ごとに金銭の異動を記録する方法が規定されている．

第３図にパターンの構造を示すクラス図を，第４図に協調を示すシーケンス図を示す．



第３図 経理パターンの構造

まず，預金が取引を行った時に，預金とコントローラが経理情報に取引の情報を書き込む．経理クラスは，取引の種類ごとにクラスメソッド（預金取引）をもっており，コントローラは取引の種類に従ってクラスメソッドを呼び分ける．経理クラスはクラスメソッドの引数として，経理情報を受け取る．呼び出されたメソッドは経理情報より変更すべき勘定科目を特定し，各勘定科目に対応するサブクラスに対してメソッド（項目修正）を呼び出す．各サブクラスは２次記憶装置からデータを読み出してその値をもつオブジェクトを生成し（準備），勘定科目の中の各項目をそれぞれのサブクラスのやり方で変更する（計算）．変更し終わったら２次記憶装置に書き戻す（更新）．



第４図 経理パターンの協調

経理パターンは，預金業務取引パターンの上に重ねて適用することにより，複数のユースケース間に共通のより詳細なアーキテクチャを提供した．

2.4.Singleton 実装パターン（実装パターン）

経理パターン中の各勘定科目オブジェクトはそのアプリケーション中で一つあればよい（Singleton パターン[2]を適用）．我々はSingletonパターンのコーディング方式を一貫させるために OOCOBOL の実装パターンを規定した．

OOCOBOL には各クラスに対してファクトリオブジェクトというオブジェクトが必ず一つ存在する．このファクトリオブジェクトは通常のクラスの定義と同様に継承関係や属性，操作を定義することができる．そこで本実装パターンではファクトリオブジェクトに Singleton となる各オブジェクトが持つ機能を実装している．

3.まとめ

本稿ではプロジェクト固有のパターンの利用，及びパターンの適用手順の規定により，アプリケーションの詳細化を効率化した事例を報告した．今後はこの考えを洗練し，プロジェクト内のパターンの体系化を検討していきたい．

参考文献

[1] 山本他 金融アプリケーション開発におけるパターン適用，オブジェクト指向シンポジウム,1997

[2] Gamma,E. 他 Design Patterns:Elements of Reusable Object-Oriented Software,Addison-Wesley,1995

[3] Bushmann,F.他 A SYSTEM OF PATTERNS,John Wiley & Sons Ltd,1996