

ソフトウェアパターンに関する現状調査

- この報告は平成13年度に(社)情報サービス産業協会が行なった「ソフトウェアパターンの実践的利用に関する調査研究」において、国内外のソフトウェアパターン専門家にヒアリングを行った結果の要約である。

調査報告書

いかにソフトウェアパターンを使えばよいか、
—情報サービス企業マネージャ—のための
ソフトウェアパターンガイドブック、2002年3月。

ソフトウェアパターンの現状と 適用経験

平成15年5月23日

三井情報開発株式会社

佐枝 三郎

従来のソフトウェア開発環境で パターンが利用されなかった原因

- ソフトウェア開発の主流であるビジネス・システム開発では、オブジェクト指向設計通用の歴史が新しい。
 - ビジネスシステムの経験者(マネージャ層)は、構造化設計、データ中心設計で育っており、オブジェクト指向設計について一部を除いてなじみがない。
 - パターンはオブジェクト思考設計の成果であり、GoFの提案から5,6年の歴史という最新概念である。
- パターン(特にGoF関連のパターン)は、デザイン上の議論であり、かつビジネスシステム経験者には難解である。
 - オブジェクト指向設計の業人には、取っ付きが悪く、概念自体が理解しにくい。
 - 事例がウィンドウシステムや、ミドルソフトウェア中心であり、ビジネスシステムとのつながりが分かりにくい。
 - 具体的にお金に結びつく、有効性を示すデータが公開されていない。

この調査の問題意識

- 問題認識:
 - ソフトウェアパターンは有用な概念であるが、わが国のソフトウェア開発現場に普及定着しているとは言いがたい。
- 背景:
 - なぜか、ソフトウェア開発現場の決定権を持つマネージャ層に、ソフトウェアパターンへの理解が少なく、現場に実践する機会が与えられていない。
- 解決策:
 - マネージャ層が、ソフトウェア開発プロジェクトでソフトウェアパターンを利用するよう動機付ける方法を見つけ、それに沿ったガイドライン資料を作成する。

ソフトウェア開発環境の変化 — パターンへの追い風 —

- ◆ ここ5年間でインターネット、WEBベースシステムが、ビジネスシステムの大きな割合を占めるよう変化した。
 - WEBシステムは、基本的にMVCのオブジェクト指向の概念に則っている。
 - ビジネスシステムでも、Java、C++、(C#)などのオブジェクト指向言語、UMLなどのオブジェクト指向設計環境を利用せざるを得ない状況。
 - 特にJavaをベースとした開発環境では、Java自身がパターンを利用して設計され、J2EEのアプリケーション開発にも、パターンの利用が推奨されている。

ヒアリング調査の対象専門家

- 海外の専門家
 - R. Johnson, L. Rising, M. Mans, M. Fowler, J. Vlissides
- 国内の専門家(研究者)
 - 青山 幹雄、中谷 多哉子、佐藤 英人、高木 浩光
- 国内の専門家(企業内の実践家)
 - 友野 晶夫、細谷 竜一、佐藤 啓太、山本 里枝子、
 - 内田 弘之

ヒアリング調査の項目

- 1 パターンを利用したシステム開発経験
- 2 利用したパターンの種類
- 3 パターン利用の動機
- 4 パターンの推進者
- 5 GoF/パターンの難しさ
- 6 パターンの適用範囲
- 7 パターンの効果
- 8 パターンの有効性の評価
- 9 パターンとフレームワーク
- 10 パターンはSE全員に必要か
- 11 パターンの教育方法
- 12 パターンを書くことについて
- 13 パターンを浸透させるための管理者の役割
- 14 企業でパターンを浸透させるには

ソフトウェアパターンの適用範囲

海外の専門家のコメント

- ・パターンはビジネスの分野でも、COBOL、VBなどOOPLでなくても活用できる。
- ・パターンの大半はパターンがよい例である。
- ・パターンはOOPLの世界だけでなく、他言語でも、ビジネスの分野でも十分に適用可能。
- ・パターンは期間活動の責務のようなもの、様々なシステムの共通性を抽出するのがパターンであるから、原則的には制限がある。現在も様々な分野で作られている。
- ・どのようなシステムにも有効であるが、特にビジネスシステムに有効なパターンが多い。

国内の専門家(研究者)のコメント

- ・デザインパターンだけでなく、具体的な範囲に対応して様々なパターンが有効。
- ・例えば性能評価パターン
- ・色々な分野で利用可能だが、重要なのはフレームワークを構築するために利用すること。
- ・基本はGoFのデザインパターン、それ以外のあたり前のノウハウをパターンというのはおかしい。

国内の専門家(実践家)のコメント

- ・デザインパターンだけでなく、分断パターンその特色々が利用可能である。
- ・フレームワーク作りに適用するのが適切であろう。但しその利用方法のためには分断パターンが必要だと考えている。
- ・人間の知的活動全て

ソフトウェアパターンの効果

海外の専門家のコメント

- ・パターンは効率をカプセル化するの理解がしやすくなる利点がある。反対に最初に理解するコストがかかる。
- ・全ての面で効果があった。パターンによるフレームワークを利用し、18ヶ月のプロジェクトが3ヶ月で完了。
- ・パターンは人間の経験を一貫化、抽象化する知的活動であるから、当然効果がある。
- ・パターンはコンポーネント化、部品化に役立つ。同時に設計者の教育ツールとして有効である。

国内の専門家(研究者)のコメント

- ・経験者のものであっても性能評価が初学者でも理解できるよになった。
- ・パターンのおよところは設計時にまわなくてよい点にある。
- ・パターンを利用しない場合にくらべ、適用したら1/5の期間で設計を終了。実装も手戻りがなく2週間で完了
- ・OOPLでの設計の基本であり、設計者の最低レベルのポキヤブラリ

国内の専門家(実践家)のコメント

- ・設計時にモジュラリティを守らせると、DBのラッピング統一化などに利用。品質面、テスト工数削減などの効果がある。
- ・生産性の効果もあるが、品質向上の効果が強い。同様なものを連続して開発、あるいは保守するのに効果が高い。
- ・パターンを利用してフレームワークを構築する結果として効果がある。実際のプログラムはフレームワークから自動生成する。
- ・開発部分を限定したため、特にバグフィックス、テストの工数が削減した。

ソフトウェアパターンの教育方法

海外の専門家のコメント

- ・受講生の真面目さに合わせて、対応するパターンを教えること。
- ・パターンは講師だけでなく、実習と自分自身はロールプレイングの方式で教えた。
- ・パターンは強調された教育にはなじまない。パターンは人間の内在的なものに依存し、改善の意欲を持たない人には教えられる。
- ・パターンは理解者が回りの人間に伝え、自分でプログラムを作って学ぶべきものである。草の根的な教育が必要

国内の専門家(研究者)のコメント

- ・意欲の差も、自分の真面目している問題に対応する具体例に基づいて学習すること。
- ・またプログラムを実験に書いて実際に動かせるべきである。
- ・パターンを利用してうまくいった事例が増え、その事例に基づいて教育するのがベスト
- ・設計者がモジュールリングを行い、自分でパターンを覚えてみるのが重要。
- ・OOPLのパターンの本質を理解させること、OOPLにおける地理的共通化、抽象化の必然であること、教育が必要

国内の専門家(実践家)のコメント

- ・基本制にCOOPPLの基礎とUMLを理解したメンバーを対象としている。
- ・自分のパターンは部門で共有しているので、新人者は関りに関するが自然と覚える。
- ・パターンを利用しないで作ったプログラムをパターンを用いて作り直すことで教育。
- ・専門家とコンセンサスを考え、レビューして、面白いと発見するところにパターンが生まれる。現場の経験を抽象化できる能力のある人を見つけてのこと。

ソフトウェアパターンとフレームワーク

海外の専門家のコメント

- ・フレームワークを作る人間にパターンは重要だが、使う人間もある程度理解する必要あり。
- ・パターンは全ての設計者のものであり、フレームワークを利用する人間も、それを直すためにパターンを知る必要あり。
- ・フレームワークは慎重に設計する必要があり、パターンが重要。アプリケーションはそれほどでもないが、保守、変更に対応するためにはパターンを理解する必要あり。
- ・デザイナーパターンは技術を高めるためのものであり、全ての設計者が理解すべきである。

国内の専門家(研究者)のコメント

- ・フレームワークとコンポーネントを作る人は詳細に理解が必要。フレームワークを利用する人は無差別に理解すればよい。
- ・フレームワーク化は重要。このプロセスもパターンを利用したテスト環境のフレームワーク化を検討している。
- ・フレームワークは多人数で体験を作って開発する場合のもの。パターンは一人で設計する場合でも有効。

国内の専門家(実践家)のコメント

- ・フレームワークは作前にコストがかかるため利用していない。パターンは知識を伝え共有するもの少数でやる場合はフレームワークは必要ない。
- ・パターンはフレームワーク構築の際の重要な手段であり、パターンはフレームワークを構築する人が理解をすればよい。
- ・パターンを理解している人は開発グループの10%程度でよい。利用者には部品を組み立てるだけである。

ソフトウェアパターンの作成

海外の専門家のコメント

- ・パターンを利用するに比べ、パターンをまとめる事は能力がいるし、時間がかかる。
- ・コンサルタントや在業のアークニカルリリーターの仕事である。
- ・パターンを書き能力を持つ人は限られている。しかし、書く人がいなければ、他の人間は使えない。それぞれの文化に適切なパターンが書かれるのを望んでいる。

国内の専門家(研究者)のコメント

- ・設計者にモデリングの時間を与える必要がある。設計者も自分のプロジェクトで問題を抽象化しパターンを抽出する試みをやら必要あり。

国内の専門家(実践家)のコメント

- ・パターンを作る人は社内、協力会社を含め、10人に一人はいる。
- ・新しいパターンを書き能力があるのは100人以上いる開発グループにうち、2、3人である。

ソフトウェアパターン浸透のための管理者の役割

海外の専門家のコメント

- ・パターンが有効だと確信し、実践する。専門組織を作るのではなく、日常に問題が起きればパターンを使って解決する、という風土を育てればよい。
- ・設計者がよいものを作りたい、勉強したいという動機を支援すること
- ・管理者は設計者を勇気づけ、新しい技術を勇躍するよう仕向ける役割がある。
- ・パターンもそのひとつである。
- ・設計者に新しいテクニックを教えさせ、勇気づけること。

国内の専門家(研究者)のコメント

- ・ソフトウェアの導入体制がフレームワーク提供と、カスタマイズ提供に別れる必要あり。そうすればフレームワーク提供者側はパターンをより活用するであろう。
- ・管理者はモデリングを行っていている設計者に余計なことを言わない、させないことで専念させるべきである。

国内の専門家(研究者)のコメント

- ・アーキテクトをリソースだと認識し、継続的に育成すること。コミュニケーションを作る環境整備。
- ・フレームワークやパターンが関係技術の維持に重要だと認識を持ってもらうこと。
- ・パターンを利用してフレームワークを作る部隊を確立すること。そしてその部隊にフレームワーク構築の分析力、構築力をつけさせること。
- ・優秀なアーキテクトを戻す、それを中心にパターンを広めること。また関係を本拠けしない社内で行い、技術の向上へのインセンティブを持つこと

調査結果のまとめ

ソフトウェアパターンの導入手順

- ・ プロジェクトメンバにパターンを考える時間を与える。
- ・ パターンを考え、他のメンバへの成果をレビュー、指導するメンタを育成する。
- ・ 社外のパターン専門家による講演など、パターンに関する最新情報が常に社内に取り込まれる仕組みを作る。
- ・ 作成されたパターンを、社内(部門内)に公開し、多くのメンバに利用してもらおうための仕組みを作る。
- ・ 優れたパターンを作成したエンジニアを勇気づけ、奨励する仕組みを作る。
- ・ 公開されたパターンの利用状況を追跡し、利用した成果と改善点をレビューできる仕組みを作る。
- ・ ソフトウェアパターンのできること、できないことをしっかりと認識する。

企業でソフトウェアパターンを浸透させるには

海外の専門家のコメント

- ・ 技術的に革新的な設計者を顕微鏡し、平均的な人間に広めること。またパターンのエキスパートコンサルタントに頼み、具体的に説明してもらうこと。
- ・ 構造的なリードを作ること。
- ・ 設計者、プログラマーに本を読み、パターンを実践する時間を与えること。また理解を手助けする人も提供する。こと。これと理解を管理者が向けることである。
- ・ 設計者にパターンを勉強する時間と報酬を与えること。パターンはよりよいデザインの知識であるから、必ず成果がある。モデルレベルが上がり、仕事のペースとなる。

国内の専門家(研究者)のコメント

- ・ オブジェクト指向設計の概念をもっと増加させる。また設計者がパターンをもっと使う努力をするべきである。
- ・ 現在のような純粋なリストだけでシステム開発を語論するのではなく、保守まで含む品質保証を含めたいビジネス形態をとっていき必要がある。
- ・ 設計者にモデリングの時間を与える必要がある。パイロットプロジェクトでパターンを実験に使う。失敗をさせずに選んでパターンによる理解させる必要あり。

国内の専門家(研究者)のコメント

- ・ プロジェクトリーダーの機能を分解してアーキテクトを役割化し、複数プロジェクトで活用すること。またトップダウンではなく、インフォーマルなアーキテクトのコミュニティを作り、皆が相談しあえる風土を作ること。
- ・ フレームワーク構築部隊と利用部隊に分けること。またフレームワークの効果を事例などで明らかにすること。
- ・ 開発を本拠に社内で行い、定期的にパターンを利用すること。実験に対象にあるパターンを興味、プログラム、テストの自動化に結びつけパターン利用を行うこと。

調査結果のまとめ(2) そのために解決をすべき課題

- ・ ソフトウェア企業が解決すべき課題
 - ・ ソフトウェアのライフサイクルを考えた、料金制度、プロジェクト運営の導入。⇒ SEにじっくり考える時間を与える。
 - ・ プロジェクトマネジメントとアーキテクト(デザイン専門家)の役割を分離する。⇒ デザイン専門家の評価を高める。
 - ・ S/Pベンダの場合に、設計・製造を大幅に協力会社に依存せず、内部で設計をしっかりと行う仕組みとする。⇒ デザイン専門家の内部育成
- ・ 研究者・専門家解決すべき課題
 - ・ ソフトウェアパターンに関する分かりやすい教科書、マニュアルなどの整備
 - ・ ビジネス分野における具体的な例題、事例集の整備