ソフトウェアパターン検索システムの構築

中山弘之[†]　鷲崎弘宜[††]　深澤良彰[†]

これまでに，ソフトウェアパターンを集積して共有を支援する様々な試みがある．ソフトウェアパターンを集積・共有する事によりソフトウェア開発における問題の解決を容易にすることができる．しかしながら異なるパターンテンプレートによって書かれたソフトウェアパターンを集めることや，集められたソフトウェアパターンから，利用者が自身の最も必要とするものを選び出すことは困難である．ここでは，既に存在するソフトウェアパターンの集積・共有を手助けする試みを紹介し，今後のソフトウェアパターン検索の発展について考察する．

Construction of Software Pattern Search Engine

Hiroyuki Nakayama,[†] Hironori Washizaki[††]
and Yoshiaki Fukazawa[†]

It already exists that accumulation and sharing of software pattern. By accumulation and sharing, the solution in the problem in software development can be made easy. However, it is difficurt that accumulation software pattern written by different pattern template and a user selects what self needs most out of the collected software pattern. In this paper, accumulation and sharing of software pattern which already exist are introduced.

1. はじめに

ソフトウェアパターンとは，ソフトウェア開発の各局面において繰り返し現れる問題に対する，考慮すべき制約を伴った解法・指針である[4)6)]．一貫した書式で記述され，複数のパターンをまとめて閲覧できるようにしたものをパターンカタログと呼ぶ．デザインパターンカタログ[7)] の発表以来，PLoP 等のパターンコミュニティの活動によりパターンの数は急速に増加し，WWW（World Wide Web）等で多数のパターンが公開されている[8)]．ソフトウェア開発にソフトウェアパターンを用いることにより，分析や設計時に頻繁に生じる問題の解決を効率的に行う事ができる．

しかし，利用者が手動で自らの必要としているソフトウェアパターンを探し適用することはあまり良い手法であるとは言えない．Web 上などの膨大な資料から，自分が必要とするソフトウェアパターンを探し出す事は困難である．また，探し出せてもそのパターンが本当に自分に必要なものかどうか，あるいはもっとも適したパターンを選んだかどうかを知る事は難しい．したがって，ソフトウェア開発者が必要としているソフトウェアパターンを検索するシステムが必要となる．これにより，ソフトウェアパターンを集積・共有する事ができれば，ソフトウェア開発は現在よりもっと効率の良いものとなるだろう．

† 早稲田大学
Waseda University
†† 国立情報学研究所
National Institute of Informatics

2. ソフトウェアパターンの共有と検索

本章では，これまでのソフトウェアパターン共有・検索の試みを幾つか紹介し，各試みの適用可能性や問題などについて考察する．

2.1 Human Based Pattern Classification (HBPC)

HBPC は，Web 上でパターンを集積・検索するシステムで，いくつかの選択からもっともその条件に適したパターンを検索する[2)]．

検索における選択は，グラフィックやデータベースといったソフトウェアパターンの分野，コード再利用性や拡張性といった必要としている機能などといったものに，normal や important といった５つの単語からなる重要度を設定して行われる．検索結果は利用者が重要度を高く設定したものに当てはまるものから表示され，pdf で書かれたそれぞれのソフトウェアパターンの文章へリンクされている．

HBPC の利点として，利用者は欲しい性質をもっ

たパターンのランク付けされたものを得る事ができるということが挙げられる．また，Web 上でパターンの集積・検索するので，誰もがいつでも使う事ができる．しかし，important 等の重要度の決定が利用者の意思に任せられるので，利用者ごとに結果が異なったり，パターンの作者と利用者間の重要度の解釈が異なる可能性がある．

2.2 A Security Pattern Search Engine

このシステムは，セキュリティーパターンをそれ固有の要素からなるクラスと属性により注釈し，検索するシステムである[1]．注釈は F-Logic Syntax と呼ばれるオントロジー言語によっておこなわれるので，パターンの形を取っていない資料（non-pattern）も注釈をつけることによりシステムで扱う事ができる．

このシステムの利点として，要素ごとの検索が行えるので，検索対象の要素を絞って検索する事ができることが挙げられる．また，パターンの形を取っていない資料も使う事ができるので，パターンの知られた利用方法と特別な状況での実地の検討がもっと簡単に行えるようになる．

セキュリティーパターンのみを検索結果として欲しい利用者にとっては，はじめから検索するパターンの領域がセキュリティーパターンのみに絞られているこのシステムは便利といえるだろう．しかしこの利点は，セキュリティーパターン以外のパターンからも結果を得たい利用者にとっては欠点ともなりえる．注釈はセキュリティーパターンの要素ごとに厳密に決められているので，ほかのパターンに適用する事はできない．

2.3 Portland Pattern Repository

Portland Pattern Repository はソフトウェアパターンのレポジトリで，検索したパターンの関係のグラフ化や，関連しているパターンの表示を行う[3]．

各ソフトウェアパターンは Wiki 上で書かれており，wiki ネームというリンクシステムを使う事により検索を容易にしている．パターンの関係をグラフ化して表示するので，利用者はパターン間関係を一目で理解する事ができる．しかし，wiki ネームを使っていない web サイトのパターンは使う事ができない．

2.4 デザインパターン利用促進のためのモデリング支援ツール

要求分析記述を解析し，適用できそうなデザインパターンを判断・提示するツールが提案されている[5]．要求分析記述自体は，自然言語で書かれており，形態素解析には CaboCha という日本語を対象としたツールを使っている．そして，形態素解析された要求分析記述は，パターンテンプレートの要素“ 目的 ”,“ 動機 ”,“ 適用可能性 ”と最もマッチするパターンを検索する．検索結果は，ユースケース図，クラス図，シーケンス図といった UML で提示する．

長所として，結果が UML の図として渡されるので理解しやすいというという点が挙げられる．また，デザインパターンに限った検索システムであり，その他のパターンには対応していないという点や．また日本語を対象としたシステムなので，英語で書かれたパターンを検索することはできないという問題点がある．

3. おわりに

本稿では，ソフトウェアパターンを集積・検索・表示する既存の手法あるいはシステムを幾つか紹介し，それぞれについて適用可能性や問題などを考察した．

従来の手法は，パターン間の関係よりも，個々のパターン内に記述された事柄を高い精度で効率よく検索することに重点が置かれている．対して我々は，パターン間の関係を事前に解析して，解析して得られた関係の強さやパターン集合の全体の構造を利用して，1 つまたは複数のパターン部分集合を効率よく検索・閲覧・利用するシステムの構築を計画している．

参考文献

1) Markus Schumacher: “Security Engineering with Patterns: Origins, Theoretical Model, and New Applications”, Lecture Notes in Computer Science, Vol.2754, Springer-Verlag (2003).
2) Vinko Vrsalovic: “Human Based Pattern Classification”, `http://patterns.ing.puc.cl/`.
3) Cunningham & Cunningham, Inc.: “Portland Pattern Repository”, `http://c2.com/ppr/`.
4) 鷲崎弘宜，深澤良彰: “ソフトウェアパターン研究の現在と未来”，情報処理学会第 141 回ソフトウェア工学研究会報告, Vol.2003, No.141 (2003).
5) 木梨充高, 小飼敬, 上田賀一: “デザインパターン利用促進のためのモデリング支援ツールの開発”，情報処理学会第 144 回ソフトウェア工学研究会報告, Vol.2003, No.144 (2004).
6) Linda Rising: “The Patterns Handbook: Techniques, Strategies, and Applications”, Cambridge University Press (1998).
7) Eric Gamma, et al.: “Design Patterns: Elements of Reusable Object-Oriented Software”, Addison-Wesley (1994).
8) Atsuto Kubo, Hironori Washizaki, Atsuhiro Takasu, and Yoshiaki Fukazawa: “Analyzing Relations among Software Patterns based on Document Similarity”, Proc. IEEE International Conference on Information Technology: Coding and Computing (ITCC2005), 2005.