議論報告: ウィンターワークショップ・イン・石垣島 「ソフトウェアパターン」セッション

鷲崎 弘宜† 細谷 竜一††

本稿では，2004 年 1 月に開催された情報処理学会ソフトウェア工学研究会ウィンターワークショップ・イン・石垣島において設置したソフトウェアパターンセッションにおける議論の内容を報告する．また，その議論の成果を踏まえて，今後パターンコミュニティにおいて取り組むべき事柄や将来の展望について述べる．

Discussion Report: Session on Software Patterns at Winter Workshop in Ishigakijima

HIRONORI WASHIZAKI† HOSOYA RYUICHI††

In this paper, we report a result of the discussion in the session on software patterns held at IPSJ/SIGSE Winter Workshop in Ishigakijima, January 2004. Moreover, we discuss the future work and prospective in patterns community based on the discussion's result.

1. セッションの目的

ソフトウェアパターンとは，ソフトウェア開発における特定の文脈上で，繰り返し発生する問題と，問題に対する解決策として得られるソフトウェア構成，および，解決にあたって考慮した制約などを包括した知識を文書などの形であらわしたものである．Christopher Alexander の建築学における 1970 年代後半の成果であるパターンランゲージ[1,2]に触発されたソフトウェアパターンコミュニティの活動の成果として，特に，オブジェクト指向に基づく分析・設計に関するソフトウェア業界の知識が底上げされた[3,4,5]．また，共有するに足ると考えられたノウハウや知識をパターン形式で記述することも一般的になりつつある．ソフトウェアに関する知識を書き表し，共有するためのソフトエアパターンという方法は一定の効果を上げたと考えられる．しかしながら，ソフトウェア設計のための包括的なパターンランゲージを志向した成果は，今にいたるまで事実上一つも得られていない．個々のパターンは孤立した存在たりえず，自ずとパターンランゲージの中の一員であることを考えると，ソフトウェアパターンのこれまでのあり方は本来的にパターンランゲージという軸を欠いている．

本セッションでは，こうした問題意識のもとで，情報処理学会ソフトウェア工学研究会パターンワーキンググループ（以降，ワーキンググループ）[6]の参加メンバと非参加メンバが合わさったグループ構成において，これまでの個々のパターンをいかに記述・利用するかという議論の一線を越え，ソフトウェアにおけるパターンランゲージとはいかにあるべきかを考える一つの起点となることを目指して議論を行った．

2. 参加者

本セッションへの参加者は，鷲崎弘宜（国立情報学研究所）と細谷竜一（東芝ソリューション）を討論リーダとして，落水浩一郎 (北陸先端科学技術大学院大学)，河合昭男 (オブジェクトデザイン研究所)，佃軍治 (日立製作所)，久保淳人 (早稲田大学)の 6 名であった．大学・研究機関と産業界からそれぞれ 3 名ずつの参加となった．議論の様子を図 1 に示す．

図 1: ソフトウェアパターンセッションの議論風景

† 国立情報学研究所
National Institute of Informatics
†† 東芝ソリューション株式会社 SI 技術開発センター
Systems Integration Technology Center, Toshiba Solutions Corporation

3. パターンランゲージ体験の事前準備

ソフトウェアパターンおよびパターンランゲージが関係する領域の大きさと多様さに関連して，参加者はそれぞれにソフトウェアパターン／パターンランゲージに対する経験や立場が大きく異なり，議論に基づく合意形成が困難であることが予想された．

そこで，議論に入る前にパターンランゲージを用いる様子を参加者全員で体験する時間を設けることを企画し，討論リーダがその準備を行った．具体的には，中埜博著『パタン・ランゲージによる住まいづくり』[7]をテキストとして，当日に読み上げる個所と，建築における問題をソフトウェア開発の文脈において置き換える際に議論すべき事柄を事前に設定した．また，パターンランゲージを活用したソフトウェア開発シミュレーションの題材を事前に設定した．

4. 議論の内容

前述のパターンランゲージ体験企画に加えて，参加者より提出された 6 件のポジションペーパ基に，以下の 3 つのテーマを設定し，ポジションペーパの発表内容を起点として各テーマに沿って集中的に議論した．

- パターンランゲージの工学的アプローチ（2 件）：ソフトウェア開発において，パターンランゲージが機能する仕組みともたらす効果を工学的に明らかにする取り組みのあり方について議論した．
- ソフトウェア開発プロセスとパターン（2 件）：ソフトウェア開発プロセスにおけるソフトウェアパターンの作用のあり方と，パターンランゲージを用いた要件定義方法について議論した．
- 組織活動とパターン（2 件）：組織活動を通じて組織とパターンがどのように相互作用するのか，そしてナレッジマネジメントのプロセスとしてパターン活動を捉えることでどのような知見が得られるのかを議論した．

5. 議論のまとめ

5.1 パターンランゲージ体験と考察

まず，前述のテキストを用いて，Alexander の理論に見られるパターンランゲージによる建築の経済性や建築物の質の問題を，ソフトウェア開発の経済性とソフトウェアの質の問題に置き換えることを試みた．テキストの読み上げと議論によって得られた知見の一部を，成果として,表 1 に示す．

表 1: 建築問題のソフトウェア問題への置き換え（抜粋）

トピック	得られた知見
建築方法の種類の違いによる問題と，ソフトウェア開発方法の種類の違いによる問題の対応	パターンランゲージの場合，前提に現在の建築は良くないというのことがある．ソフトウェアにも対応する問題はある．建売住宅とは，パッケージソフトウェアの適用に相当する．いらない機能ごと買わされる顧客にフィットしていないといった問題を持つ．増改築は，ソフトウェア拡張に相当する．ソフトウェアでは拡張を繰り返すと内部設計が複雑となり保守が困難となる．
ソフトウェア開発におけるアーキテクトビルダ	アーキテクトビルダはソフトウェアにおいても理想的な存在だが，ソフトウェア業界の開発体制上，設計もできる人が一貫してユーザ要求を聞き，設計を指揮するということはできない．つまり体制上の課題がある．
ソフトウェアの形を決めるルールと美しさ	ソフトウェアの場合，パターンを適切に展開した設計が美しい設計となるのか，不明である．
ソフトウェア開発における「家相」	「家相」はパターンを繰り返し適用した結果全体の一貫性を獲得する上で重要な考え方である．ソフトウェアでは，アーキテクチャが家相に相当する．
ソフトウェア開発におけるストーリー	アーキテクトビルダーやイテラティブな設計，ストーリを中心においた要求獲得など，エクストリームプログラミング（XP）の着想がパタン・ランゲージから借りてきたものではないかという推測を支持している．しかし XP ではパターンの話はこれといって出てこない．少数の熟練した開発者によるプロセスを前提にしているからと推測される．
ソフトウェア開発における要求獲得とパターン獲得	一戸建て住宅建設の場合，ビルダーは家族の代表と交渉しながら設計する．ここでパターンランゲージを使って要求を引き出してる．こうして合意形成に向けたプロセスが始まっていく． ソフトウェア設計の場合でも，SE または設計者がパターンランゲージ（的な知識）を持っている場合と

	そうでない場合とで要求抽出プロセスの質は変わってくるだろう．ソフトウェア開発の場合，顧客側の窓口の言っていることだけを聞いていてもあとで覆ることもあるが，かといってすべてのステークホルダーに接触して要求を引き出すことも難しい．
ソフトウェア開発において「現場」が与えるもの	ソフトウェア開発では，開発したシステムを設置して運用する実環境，実環境をとりまくビジネスプロセス，アクター（顧客）などが現場に相当する．設計上動く，あるいは開発環境上動くことと，実環境で動き，うまく運用されることは必ずしも一致しない．「現場検証を行うべきである．

続いて，Web アプリケーションのユーザインタフェース（UI）設計を題材として，要求獲得を対話形式で行い，要求を解消するパターンをその場で抽出し，複数のパターンの適用順序を考案してパターンランゲージを編み上げることを試みた．

まず我々は，パターンランゲージを発見する手順を以下のように仮定した．

(1) 求める「名の無い質」（QWAN）を記述する．
(2) QWAN を得ようとする際に生じるフォース（非機能要件を満たすために生じる制約）を識別する．
(3) フォースを解消するための個別の方策（パターン）を挙げていく．
(4) 方策の集合を階層化し，階層間・方策間の関連付けを行うことでパターンランゲージを得る．

この仮定に基づいて，アーキテクトビルダ役（A）と顧客役（C）が以下のような対話を行った．

A: どういうものが欲しいのか？
C: 使いやすいものが欲しい．
A: どういうものが具体的に使いやすいのか？
C: 経理のようなエキスパートは画面をほとんど見ずに効率よく使えるもの．さらに，初心者でも扱えるもの．
A: では，経験者用と初心者用の 2 つのモードを提供することにしよう．初心者用の UI をイメージして欲しい．
C: ウィザード方式で，1 手順 1 画面がいい．
A: そこには，どういった使いやすさが求められるのか？
C: 反応の早いこと，処理を取り消し可能，作業を中断可能であることなどが挙げられる．
A: では，一つずつ解決していこう．ここに，UI 憲法と呼ばれる UI に関するパターンランゲージがある．解決に使えそうなパターンを選択して，対応付けていこう．

（以降，UI 憲法中のパターンと非機能要件の対応付け作業を行った．）

上述のように，体験シミュレーションでは，非機能要件を列挙した後で，非機能要件に対応するパターンを，トラック参加者が作成していた UI 憲法（河合昭夫著『EXOS 統一操作環境の実現』[8]）から選び出して対応付けることによって，その非機能要件から生じる UI 設計上の制約を解消することを試みた．対応付けと，予想される解決の一部を表 2 に示す（全体は本稿末尾の付録を参照）．時間の都合上，各解決を用いる順序を決めることができなかったが，得られたパターンと解決の集合はパターンランゲージの雛型になっていると考えられる．

表 2: 非機能要件と UI 憲法の対応付け（抜粋）

非機能要件	UI 憲法の条文（パターン）	条例（パターンが与える解決）
反応の遅さを感じさせない	第 6 条：必ず即座に反応	JavaScript などによるクライアント側での処理機能
取り消しができる	第 7 条：逆操作を許す	Undo 機能
作業を中断・再開できる	（対応する条文無し）	一時保存機能
間違えたからといって，利用者の落ち度にしない	第 8 条：簡単なエラー処理	能動的なエラー訂正支援機能

この体験から，次のような結論を得た．まず，フォースを明示的に記述することはできなかった．フォースの見極めは，扱う領域に熟練していなければ困難である．また，UI 憲法のような，扱う領域に関するパターンランゲージを用意しておくことについて以下の知見が得られた．

- 用意されたパターンランゲージに基づいて，状況に特化したパターンランゲージを編み上げられる．
- 顧客が提示していなかった非機能要件を，用意されたパターンランゲージから逆に推測して提案し，顧客満足度を向上させることができる．例えば，条文第 8 条に相当する非機能要件は，顧客は提示していなかったが，アーキテクトビルダからの提案により必要である．
- 既存のパターンだけでは不十分であることがあり，他の開発経験などに基づいて，その場で新たにパターンを作る能力が必要である．例えば，「作業の中断・再開」に関するパターンは，UI 憲法中には存在しなかった．
- UI は視覚的イメージを共有できるため，パターン

ランゲージを作成しやすい．

5.2 パターンランゲージの工学的アプローチ

これまでのソフトウェアパターンへの工学的な取り組みのほとんどが，デザインパターンの扱いに関する手法（特にデザインパターンの適用手法・検出手法）であり，他のパターンへの展開が不足している現状を確認した．鷲崎らの調査によれば，パターン研究に関する主要な 41 の文献のうち，デザインパターン以外のパターン（アーキテクチャパターンなど）に関するものはわずか 9 件であった（鷲崎ほか『ソフトウェアパターンランゲージ工学に向けて』[9]）．

このような現状において，パターン形式がもたらす効果に関する調査，および，認知科学・言語理論・言語学といった他の学問領域との学際的な取り組みが，今後の重点的に取り組むべき課題であるとの結論に達した．

パターン形式とパターンの関係については，久保らによるパターン文書解析の試みの発展が期待される（久保ほか『ソフトウェアパターンの自動的な体系化の試み』[10]）．この試みでは，テキスト処理によってパターン文書のパターン形式を自動的に判別し，パターン文書間の類似度に基づいた関連付けを試みている．

学際的な取り組みについては，まず，パターンは知識を表すため，認知科学における認知の仕組み／表現方法と，パターンランゲージにおける要求獲得／パターン表現の関係について明らかにする必要がある．また，パターンランゲージにおける言語理論上の語と文法を明らかにすることで，パターンランゲージがランゲージ（言語）としての生成的特性を有するかどうかを明らかにする必要がある．

5.3 ソフトウェア開発プロセスとパターン

ソフトウェアパターン技術が，これまでのソフトウェア工学体系において他の技術と結合しておらず，効果を発揮しきれていないことを確認した．

その解決にあたり，次の 2 つの取り組みが重要であるとの結論に達した．

まず，既存のソフトウェア開発方法論／プロセスと，ソフトウェアパターンとの統合が求まれる．その試みとして，落水は，ロール（関与者の役割）に基づいて RUP に Coplien らの組織パターンを組み込みを行っている（落水『ソフトウェア開発方法論とソフトウェアパターン』[11]）．今後の，実験的な評価などが期待される．

また，要求工学の観点からパターンランゲージがもたらす役割と効果を明らかにすることが重要である．河合の経験と報告によって，非機能要件の獲得と記述にパターンランゲージが有効に機能するとの結論を得た（河合『パターン言語を非機能要件定義に』[12]）．非機能要件は，定義するものではなく，パターンランゲージを用いて自然に解消するものである．従来の要求工学では，非機能要求はチェックリストとしてまとめておく方法をとる．対してパターンランゲージは，パターンのタイトルのリストそのものはチェックリストと差異は無いが，生成能力が大きく異なるとの結論に達した．扱う問題が巨大であるとき，取りうる解をチェックリストでは網羅することが困難である．対して，パターンランゲージでは少ないパターンの組み合わせによって最適な解を生成する．

5.4 組織活動とパターン

これまでのパターン活動が，パターンを書くことにのみ目を奪われており，各パターンが気の利いたメモ書きとさしてかわらない現状を確認した．

組織活動においてパターンを生かすための方策として，佃は，ナレッジマネジメントにおける表出化・連結化・内面化・共同化から構成される一連の知識運用サイクルが，パターンの運用についても同様に重要であることを指摘している（佃『パターン活動とナレッジマネジメント』[13]）．この指摘に基づいて，パターンは書いたら終わりではなく，日常的な活動としなくては成らないことをナレッジマネジメントから学べることを確認した．

また，そのようなパターン運用の全体的な枠組みが必要であることを確認し，その土台として，細谷が提案するパターンエンジン（細谷『知識創造装置パターンエンジン』）を利用可能であるとの結論に達した．パターンエンジンは，パターンを知識の媒体として捉えて，知識の発生や共有・進化といったナレッジマネジメントの過程をモデル化した仮想の装置である．パターンエンジンを用いることで，パターンランゲージとして記述されたパターンライフサイクルマネジメントを実践し，パターンを組織活動に役立てられる可能性を確認した．

6. パターン宣言

以上の議論を踏まえて，我々は会議の終わりに，「パターンはソフトウェア技術者・組織の資質を磨くために不可欠である」と高らかに宣言した．この宣言を，パターン宣言と呼ぶことにする．パターン宣言は，パターンがコミュニケーションの質の向上と，個人の良識を養うことに有効であるとの考えに基づいている．

パターン宣言を締結するために，今後において具体的な取り組むべき方策を以下に挙げる．

（1）有用なパターンの提示:UI に関するパターンランゲージの事例研究を進めると同時に，他のパターンランゲージ化に適した問題領域の探索が必要である．

（2）パターン運用の枠組みの確立：実行可能なパターン活動支援技術を確立・普及させる必要がある．また，開発プロセスとパターンランゲージのさらなる融合化が求まれる．

（3）パターンに関する基礎理論の探求：前述の重点課題を解決することによって，パターンランゲージ工学を確立することが求まれる．

7. 将来の展望

今後は，パターンワーキンググループが主体となり，上述の各方策への取り組みを行い，その具体的な成果を広く公開・共有すると共に，ワーキンググループ内外の実務家・研究者間で議論と実践を行い，ソフトウェア開発活動および組織活動一般におけるパターンランゲージの活用を目指していく．すでに，以下の取り組みを開始している．

方策（1）については，ワーキンググループの勉強会を起点として，Web の UI デザインに関するパターンランゲージの事例調査と，新たなパターンランゲージの作成を開始している（河合『UI パターン言語の試行』第 3 回パターン勉強会，2004）．

方策（2）については，個人レベルでの研究に加えて，ワーキンググループ Web サイトにおいてパターンライフサイクルマネジメント・パターンランゲージの初期版を公開し，その議論と洗練を開始している．

方策（3）については，ワーキンググループ研究タスクにおいて，複数の大学・企業研究者によってパターン研究動向のさらなる調査と，パターン関連ツール／手法と現実のパターン活動の対応付けを開始している．

参考文献

[1] Alexander, Christopher. *The Timeless Way of Building*. Oxford University Press, 1979.

[2] Alexander, Christopher, et al. *A Pattern Language*. Oxford University Press, 1977.

[3] Beck, Kent and Ward Cunningham. “Using Pattern Languages for Object-Oriented Programs.” *Position statement for the OOPSLA’87 workshop on the Specification and Design for Object-Oriented Programming*, 1987. http://c2.com/doc/oopsla87.html

[4] Gamma, Erich, et al. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[5] 本位田真一 and 吉田和樹(監訳). *オブジェクト指向による再利用のためのデザインパターン*. ソフトバンクパブリッシング, 1995.

[6] 情報処理学会ソフトウェア工学研究会パターンワーキンググループ, http://patterns-wg.fuka.info.waseda.ac.jp/

[7] 中埜博. *パタン・ランゲージによる住まいづくり*. 井上書院, 1988.

[8] 河合昭夫. EXOS 統一操作環境の実現. In *UNISYS 技報*, 1991.11.

[9] 鷲崎弘宜 and 深澤良彰. ソフトウェアパターンランゲージ工学にむけて. In *情報処理学会ウィンターワークショップ・イン・石垣島論文集*, 2004.

[10] 久保淳人 and 鷲崎弘宜 and 深澤良彰. ソフトウェアパターンの自動的な体系化の試み. In *情報処理学会ウィンターワークショップ・イン・石垣島論文集*, 2004.

[11] 落水浩一郎. ソフトウェア開発方法論とソフトウェアパターン. In *情報処理学会ウィンターワークショップ・イン・石垣島論文集*, 2004.

[12] 河合昭男. パターン言語を非機能要件定義に. In *情報処理学会ウィンターワークショップ・イン・石垣島論文集*, 2004.

[13] 佃軍治. パターン活動とナレッジマネージメント. In *情報処理学会ウィンターワークショップ・イン・石垣島論文集*, 2004.

[14] 細谷竜一. 知識創造装置パターンエンジン. In *情報処理学会ウィンターワークショップ・イン・石垣島論文集*, 2004.

付録: ユーザインタフェース設計における非機能要件とUI憲法の対応付け