

特別講義録フォーマット

第 6 回特別講義 レポート

日時	2021年10月8日(金) 10:00~12:00
実施形態	オンライン (Zoom) 開催
テーマ	パターンと品質
講師名・所属	原田 騎郎 氏 (株式会社アトラクタ Founder 兼 CEO)
司会	永田 敦 氏 (サイボウズ株式会社 / 本研究会 研究コース 4 主査)
アジェンダ	<ol style="list-style-type: none">1. パタン (パターン) とは何か?<ul style="list-style-type: none">・ パタン、パタンランゲージの背景2. ソフトウェア開発におけるパタンの活用<ul style="list-style-type: none">・ 設計パターンとして・ 組織パターンとして・ パタンランゲージとしてのスクラム3. プロダクトにおける品質の課題4. 品質保証からアジャイル品質への変化<ul style="list-style-type: none">・ 利用者品質をパターンで記述してみる・ 品質ゲートから継続的な品質管理活動へ5. スクラムによる開発での適用の試み6. まとめ
講義の要約	
<p>◆ 講師紹介 原田 騎郎 (株式会社アトラクタ Founder 兼 CEO) アジャイルコーチ、ドメインモデラ、サプライチェーンコンサルタント。 認定スクラムプロフェッショナル。 外資系消費財メーカーの研究開発を経て、2004 年よりスクラムによる開発を実践。ソフトウェアのユーザーの業務、ソフトウェア開発・運用の業務の両方を、より楽に安全にする改善に取り組んでいる。認定スクラムトレーナー (CST-R) 。</p> <ol style="list-style-type: none">1. パタン (パターン) とは何か?<ul style="list-style-type: none">● クリストファー・アレグザンダーの問題意識	

アレグザンダーはオーストリアのウィーン出身の建築家。

第2次世界大戦後、復興のために建てられる建築物は、人が住むことや人がどう動くか考慮されておらず質が悪かった。アレグザンダーは、そのような建築物では人がいきいきとしていないと考えた。

- アレグザンダーが初期にまとめた3部作

- オレゴン大学の実験（1975）

オレゴン大学のキャンパス再建の際に、学生や教授を意思決定に参加させた実験について書かれている。

アレグザンダーは建築において、作る人と使う人が分断されてしまうことで、使う人が本当に欲しいもの、使いやすいものにならず、使う人自身がうまく直せないものになってしまうという問題意識を持っていた。そのため、建築家が要求通りに建てるだけではキャンパスを使う人がいきいきしないと考えた。

学生や教授に参加してもらうことで、キャンパスをより良くするための設計要素、解決策がいくつか挙がった。

例) 駐車場を1箇所に作らない

歩道を交わるようにする

- パターンランゲージ（1977）

キャンパス、ビル等建物を建てる時に、どのようなことを考えたらうまくいったかという例を集めたもの。253もの例が挙げられている。

- 時を超えた建設の道（1979）

作られた建築が長持ちするためには、どんな特性が必要か書かれている。

- パタンとは何か

1つ1つのパタンは、身の回りで起きた問題とそれに対する解決策を示したもの。

状況が変化した時にも見た人がすぐに使えるように具体的に記述される。

パタンは、何回適用しても同じように起こる問題を解決してきた実績があるが、まったく同じ状況は無いため、適用方法は少しずつ違うはずである。

例 1) 154. Teenager's Cottage

問題：子離れ

解決策：敷地の中に子供が住むための小さい家を作る

例 2) 155. Old Age Cottage

問題：親の面倒を見る

解決策：敷地の中の小さい家に親が住む

パターンにはシーケンスという考え方があり、パターンランゲージでは、うまくいった事例でのパターンの適用順についても説明される。

例) 154. Teenager's Cottage の後に、155. Old Age Cottage を適用することで、建物を長持ちさせることにつながる。

- パターンランゲージとは？
 - 順序だって適用できるパターンの集まり
 - パターン一つ一つが全体の質をよくする
 - まとまって、新たなパターンを生み出す可能性がある

結果的に無名の質ができ、心地よくいきいきとしている状態が達成される。

2. ソフトウェア開発におけるパターンの活用

大規模開発が広がる中で、ソフトウェアにもパターンを適用することができないか考えられるようになった。初期の段階では、ケント・ベックやウォード・カニンガムらが WikiWikiWeb という WEB サイトで、ソフトウェア開発においてうまくいった話をドキュメント化し、共有した。また、PLoP (パターンについて話し合う活動) が行われるようになり、様々な成功したパターンが集められるようになった。

- 設計パターンとして
 - デザインパターン (Gang of Four、1994)
どのような設計、プログラムだと作りやすいか、見通しが良くなるかについて書かれている。
 - アナリシスパターン (Martin Fowler、1996)
どういう形でまとめておくと、ソフトウェアとして扱いやすくなるか書かれている。ある程度大きな規模のソフトウェアを作る場合には、有用なパターンとなる。
 - Extreme Programming Explained (Kent Beck、2004)
うまくいったソフトウェア開発のパターン (プラクティス) の集まり。技術に関するものだけでなく、よいプログラマーになるための習慣について書かれている。
 - ドメイン駆動 (Eric Evans、2003)
アレグザンダーが目指した「作る人と使う人が一緒になって進める」という考え方をソフトウェアにもっていく方法を主張している。

- エンタープライズアプリケーションパターン (Martin Fowler、2002)
大規模システムをどうやって扱うべきか書かれている。
- Enterprise Integration Patterns (Gregor Hohpe & Bobby Woolf、2003)
大規模システムをどのようにつなぐか、どのようにつなぐことでうまく作ることができるか書かれている。現在のマイクロサービスの考え方のもとになるもの。
- 組織パターンとして
 - Development Process Generative Pattern Language (1995)
組織構造やプロセスにもパターンが適用できることが書かれた論文。
 - Fearless Change (Mary Lynn Manns, Ph. D、 Linda Rising Pd. D、2004)
より良いソフトウェアを作るための組織づくりや組織自体を活発にするためのパターンが書かれている。
例) 食べながら話をする。
 - 組織パターン (James Coplien & Neil B. Harrison、2004)
ソフトウェアをうまく作れる組織の形がパターンとして書かれている。
- パターンランゲージとしてのスクラム
組織の作り方やうまくいったプロセスを集めて書かれたものがスクラムのもとになっている。スクラムはパターンランゲージとして書かれた。
- A Scrum Book - The Sprint of the Game (The Scrum Patterns Group、2019)
スクラムについてパターンランゲージとしてまとめたもの。
- アジャイル開発宣言について
当時、大規模開発ではソフトウェアファクトリーが主流だったが、ソフトウェアファクトリーでなくても、パターンを集めることでうまく開発ができることが分かっていった。そして、作る人と使う人が一緒になって進めるほうが質の高いものができると考えられるようになり、そのように考えるコミュニティでの合意事項がアジャイル宣言となった。

3. スクラムによる開発での適用の試み

- Product Organization Pattern Language
<プロダクトオーナーに関するパターン>
Product Owner

問題：ステークホルダーが増えるほど、判断に時間がかかる

解決策：最終判断を一人が行う

↓

Product Owner Team

問題：プロダクトオーナー一人がすべての判断を行うため、負荷が高くなる

解決策：プロダクトオーナーを支えるチームを作る

- Value Stream Pattern Language

<スプリントに関するパターンの一部>

Sprint

↓

Product BackLog

解決策：プロダクトバックログを作る

↓

Refined Product BackLog

解決策：選ばれたバックログを作る

- スクラムにおけるプロダクトの品質

スクラムには、出来上がったプロダクトの質を常に早く確認するための仕掛けがある。

- Regular Product Increment

プロダクトの増分をステークホルダーに見せて、価値を常に確認する。

- レトロスペクティブ

内部の設計やプロセスの質を確かめて、プロダクトと答え合わせをする。

4. 品質保証からアジャイル品質への変化

- QA2AQ について

QA が開発部門とは別の部門にある場合の組織パターン。今後、QA が全体として質を考える組織に変わっていくために必要なパターンランゲージが記述されることが望ましい。

- 利用者品質をパターンで記述してみる

パターンを使って、品質をより良くするためにできることが記述されるようになってきた。
例) 使う人のパターン (ユーザーインターフェース)

スクラムでは、ステークホルダーを巻き込んでレビューを実施し、プロダクトの価値を確かめるプロセスがあるが、良いプロダクトとは何か、利用者にとって良いプロダクトの作り方は何かについて、パターンで記述されていない。

今後、ユーザーが使いやすいパターンが記述されれば、それらのパターンをもとに、開発者やオーナーがより良いパターンについては話合えるようになる。

5. まとめ

ユーザーにとってよいプロダクトとなるパターンを書き、集めることが今後の展望。
パターンに基づいて作ったプロダクトのテストができる人は、パターンがうまくいったか確かめることができるはず。うまくいったパターンを書き残しておくことが望ましい。
定性的なものマトリクスの両方があることは、ソフトウェア開発にとって大きなメリットとなる。

6. 質疑応答

<質問>

「パターンを集めるときは全く同じものはない」ということは、パターンを集める際には似たものを集めていくのか。

<回答>

パターンのコミュニティが使っている方法は、KJ法に近い。カテゴリで集めるというよりは、それぞれのパターンが自然に集まってくるのを待つ。新しいパターンを作るのではなく、みんなが知っていることに名前を付ける。

<質問>

パターン≠使える事例の集積、ということであるか。

<回答>

うまくいくやり方が集まったもの。うまくいく順番や組み合わせを含めて書かれたのがパターンランゲージ。

<質問>

パターンとテスト設計は似ているか。

<回答>

似ている。「xUnit Test Patterns」という本があり、テスト設計もパターン集がある。

<質問>

KPIなどで振り返りした結果が組織としてのパターンになる可能性があるか。

<回答>

KPIもパターンに出てくる可能性もある。KPI自体もパターンと考えられる。

<質問>

パターン≠プロセスになってしまうが、パターンについてよい説明はないか。

<回答>

パターンの中にはプロセスとなっているものもある。

パターンは何度も使えるものである必要がある。うまくいかないパターンが出てくることがあるが、どこを改善すればうまくいくか考え、修正すればよい。スクラムパターンの本に出てくるパターンは、少なくとも3回以上書き直されている。

<質問>

パターンとモデルの違いは何か。

<回答>

モデルは対象物と関係する。パターンは状況に対する扱い方である。パターンは、コンテキストがあてれば対象物が違ってもうまくいくことがある。モデルとしてパターンを書くのもよい。パターンとして正しいというよりも、パターンとして有効、役立つものであればよいのではないか。

<質問>

パトレットを公開し、うまくいったか共有するようなコミュニティがあれば教えてほしい。

<回答>

慶應義塾大学の井庭先生は、学習、プレゼンのパターンを集めている。鷲崎先生は、ソフトウェアに関わるパターンを集めている。本となって普及する前のパターンは様々なものがあるので、探してほしい。

以上