ユーザーストーリー：ファースト・ジェネレーション

Scrum Gathering Tokyo 2011

2011年10月22日

ワイクル株式会社 取締役

角 征典 a.k.a. kdmsnr

kado.masanori@waicrew.com

本日の想定する参加者

- ソフトウェア開発の関係者である
- スクラムの基礎知識がある
- ユーザーストーリー を使ったことがない／うまく使えない

Q.「対象外の人はいますか？」

対象外であっても楽しんでいってください＞＜

角 征典 - kdmsnr

http://www.slideshare.net/SukusukuScrum/no01101suc3rum20100225

角 征典 - kdmsnr


スクラムガイド
Scrum.org


FrontPage
Martin Fowler's Books


アジャイルレトロスペクティブズ

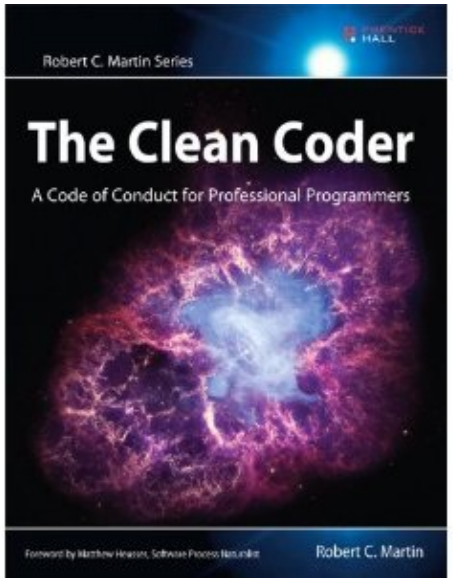

ウェブオペレーション


メタプログラミング
Ruby
メタプログラミングは
魔術みたいなもので、
ビックリすることが
できるようになる。
——Matz


Facebook
マーケティング

角 征典 - kdmsnr

鋭意翻訳中!!

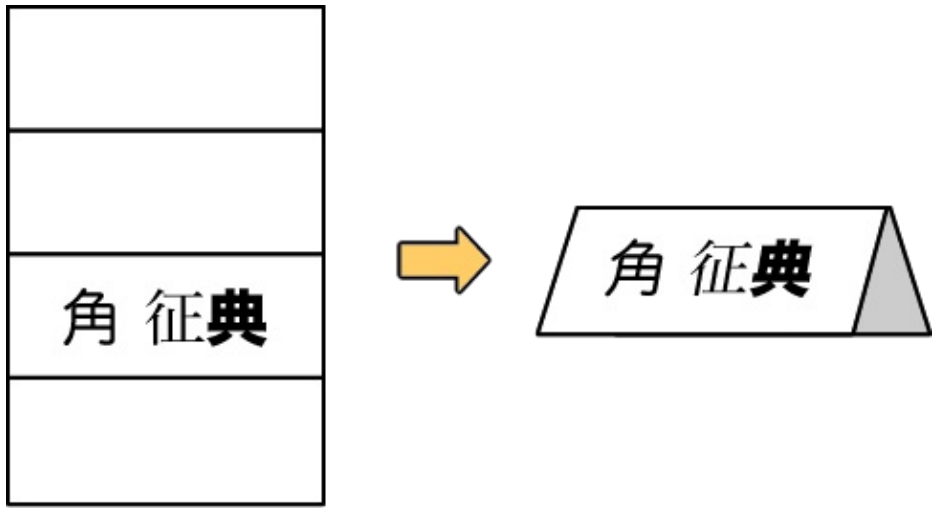


グループづくり

- できるだけ知らない人同士で
- うまく「多様性」ができるように
- n人のグループを作ってください

ネームプレートづくり(5分)

A4用紙でネームプレートを作ってください



1人ずつ自分の名前とその文字を紹介して、
グループの人に1人1文字ずつ書いてもらってください。

※ 本名が嫌ならIDやニックネームでもOK
※ 文字数や人数が足りなければ、適宜調整してください

これからお話すること

目標「ユーザーストーリーの 思考方法を身に付ける」

- 第1部：ユーザーストーリーの 物語
- 第2部：実例による仕様
- 第3部：パターン、リーン、ユーザーストーリー

10分経過

13:30

残り110分

第1部
ユーザーストーリーの物語

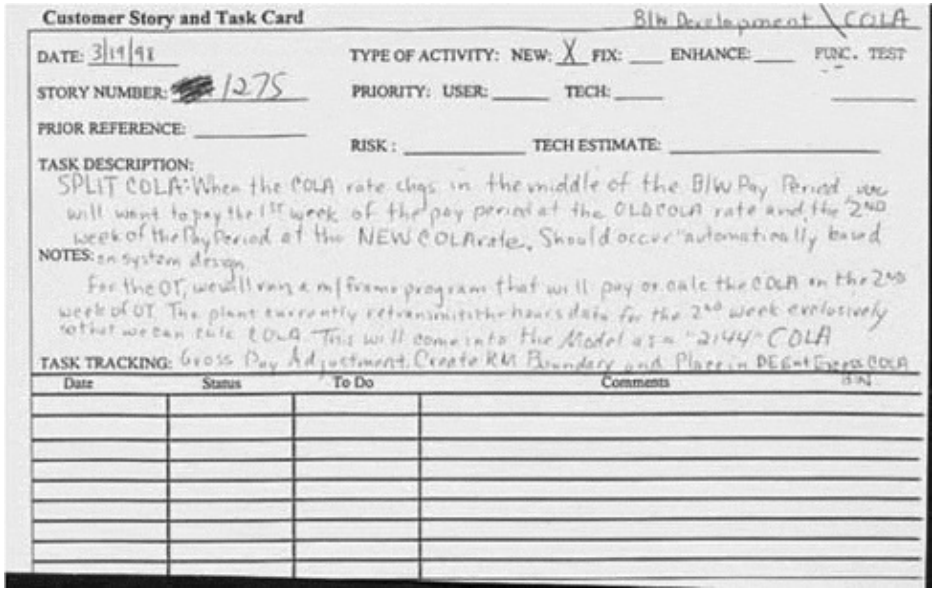
石のスープのおはなし

『せかいいち おいしいスープ』（マーシャ・ブラウン）
http://www.amazon.co.jp/dp/4001112175

ユーザーストーリーのスープ

- それ自体は大したことがない
  - 要求を網羅した仕様書ではない
- 会話のきっかけとなるもの
  - みんなで情報や一時成果物を集める
- 最初から最終成果物はわからない
  - 創発的（emergent）なものである

XPのストーリーカード 



Customer Story and Task Card — B/W Development \ COLA

DATE: 3/19/91 — TYPE OF ACTIVITY: NEW: X FIX: ___ ENHANCE: ___ FUNC. TEST

STORY NUMBER: [redacted] 1275 — PRIORITY: USER: ___ TECH: ___

PRIOR REFERENCE: ___ — RISK: ___ TECH ESTIMATE: ___

TASK DESCRIPTION:
SPLIT COLA: When the COLA rate chgs in the middle of the B/W Pay Period we will want to pay the 1st week of the pay period at the OLD COLA rate and the 2nd week of the Pay Period at the NEW COLA rate. Should occur automatically based on system design.

NOTES:
For the OT, we will run a mainframe program that will pay or calc the COLA on the 2nd week of OT. The plant currently retransmits the hours data for the 2nd week exclusively so that we can calc COLA. This will come into the Model as a "2144" COLA

TASK TRACKING: Gross Pay Adjustment, Create RM Boundary and Place in PEG at Execute COLA

Date	Status	To Do	Comments	B.N.

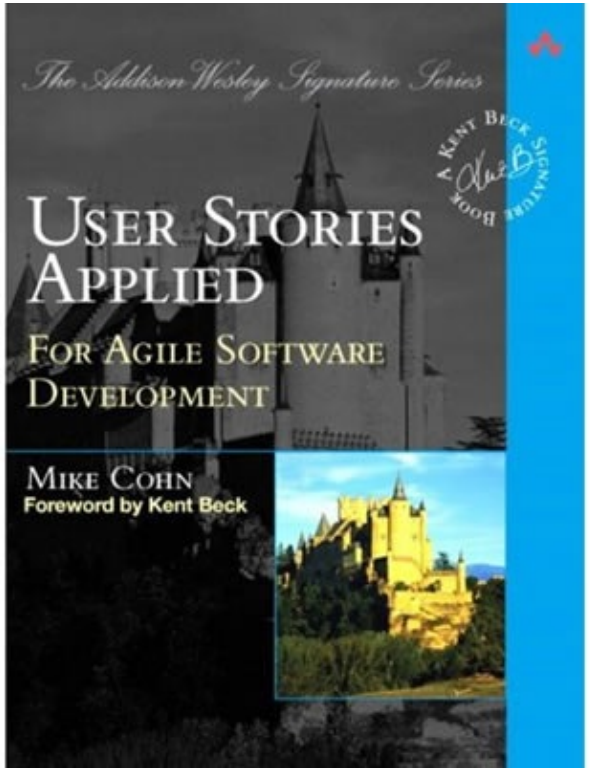
『Extreme Programming Explained: Embrace Change』（Kent Beck）1999年

くお話を聞かせてください)

- 開発者が要求を「記述」していてはうまくいかない。
- 要求の「自由」と「責任」をビジネスと分担するのがいいと思う。
- ビジネスが積極的に関わってくれるような「形」が大事だ。

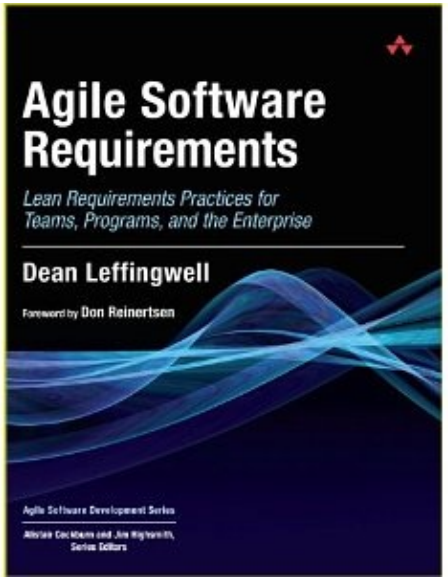
http://c2.com/cgi/wiki?UserStoryAndUseCaseComparison

ユーザーストーリーの手法化



『User Stories Applied: For Agile Software Development』（Mike Cohn）2004年

アジャイルソフトウェア要求



『Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise』（Dean Leffingwell）2011年

ユーザーストーリーって何？

> “ユーザーや顧客のシステム要求が完了できるように、関係者全員がわかる言葉で理解していくためのもの。
> ―角 征典

※なかなか良い定義がなかったから自分で定義した!!

ユーザーストーリーって何？

- (1) システム要求が
  - ビジネスとソフトウェアの間（海面）
- (2) 完了できるように
  - 終わりが見えなければいけない
- (3) 理解していくためのもの
  - 記述ではなく段階的に共有理解する

これユーザーストーリー？

理由も一緒に話し合ってください（8分）

1. アンドゥは50回まで
2. 10/22までにシステムを利用可能にする
3. 画面遷移なしでカートに商品を追加できる
4. Ruby on Rails 3.1で開発する
5. JSONでデータを出力する
6. 出荷済商品を検索できる
7. システム導入後の売上を1.5倍にする
8. VOCの数値を入力する

これユーザーストーリー？

「場合による」ことが多いけど……。

1. △ アンドゥは50回まで
2. ✘ 10/22までにシステムを利用可能にする
3. △ 画面遷移なしでカートに商品を追加できる
4. ✘ Ruby on Rails 3.1で開発する
5. △ JSONでデータを出力する
6. ◯ 出荷済商品を検索できる
7. ✘ システム導入後の売上を1.5倍にする
8. △ VOCの数値を入力する

25分経過

13:45

残り95分

三幕構成 by connextra

1. <役割>として、
2. <対象>を（に）<行為>したい。
3. それは<価値>のためだ。

参考：『アジャイルな見積りと計画づくり』
http://www.amazon.co.jp/dp/4839924023

これは覚えて帰ってください!!

三幕構成は「物語の形式」

物語を書くようにユーザーストーリーを書く

1. 設定（誰がどんな状況なのか）
2. 葛藤（何ができないのか）
3. 結末（何が欠かせないのか）

『映画を書くためにあなたがしなくてはならないこと』
http://www.amazon.co.jp/dp/4845909278

たとえば、映画「ロッキー」

1. 設定：ペットショップの店員に恋する三流ボクサーとして、
2. 葛藤：世界チャンピオンとのタイトルマッチに挑戦したい。
3. 結末：それは自分がゴロツキではないことを証明するためだ。

参考：http://ja.wikipedia.org/wiki/ロッキー_(映画)

設定を探せ（2分）

sgt2011に来てるのはどんな人？

まずは自分の設定をカードに書いてみよう。

> （例）スクラムに興味があるマネージャとして、sgt2011に行きたい。

設定を作るヒント

- 既知の事実（自分や知人から）
- 観察やインタビュー
  - Contextual Inquiry
  - 師匠と弟子モデル
- 葛藤から仮説→調査
- 勝手な想像でペルソナを作るくらいなら、実在の人物を設定する

結末を探せ（5分）

sgt2011に行った結末は？

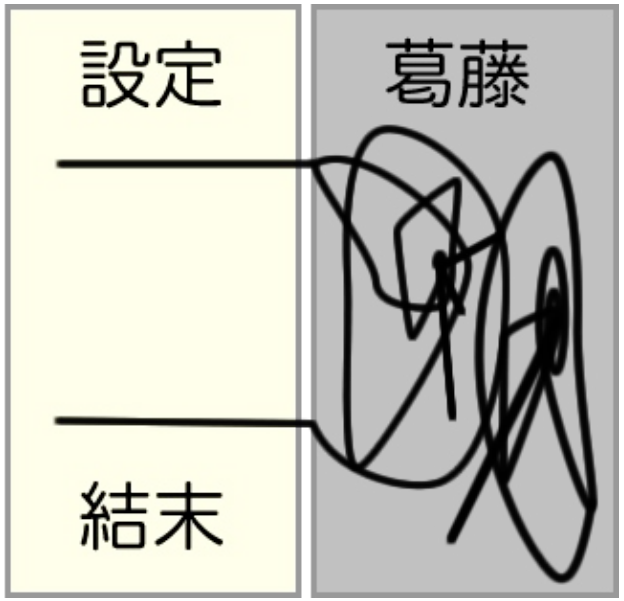
まずは自分の結末をカードに書いてみよう。

続いてグループで自分以外のストーリーを書いてみよう。

> （例）スクラムに興味があるマネージャとして、sgt2011に行きたい。
> それは実際の導入事例を知るためだ。

設定と結末を大切に

葛藤はいろいろあるけれど、



物語はすべて「行きて帰りし物語」である。

（参考）価値を重視した記法

フィーチャーインジェクションテンプレート

1. <価値>のために、
2. <役割>として、
3. <対象>を（に）<行為>したい。

45分経過

14:05

残り75分

良いユーザーストーリーとは？

INVEST by  Bill Wake

[I]ndependent - 非依存
[N]egotiable - 交渉可能
[V]aluable - 価値がある
[E]stimatable - 見積り可能
[S]ized Right - 適切な大きさ
[T]estable - テスト可能

次の計画づくりに使えるかどうかが大事

良いストーリー（物語）とは？

続きが気になるもの

http://ja.wikipedia.org/wiki/シェヘラザード

もっと詳しく（8分）

- 1人が自分のストーリーを発表して
- みんなでオープンクエスチョン*をしてみましょう。
- 時間が余れば、次の人が自分のストーリーを発表してください。

[*] 答えがYES/NOにならない質問

ここまでのまとめ

ユーザーストーリーは、

- 石である
- 物語のように**三幕構成**で書く
- 続きが気になったら**質問**する

55分経過

14:15

残り65分

第2部 実例による仕様
Specification by Example

3行でいいの？楽勝www問題

- いい時もあれば悪い時もある
- 「記述」が最終目標じゃないよ！
- 「横」が決まったら次は「縦」

INVEST by  Bill Wake

- テスト可能であること
  - 完了や受入が可能であること

「テスト」って言葉が微妙すぎ

原因はもちろん ⇒ 

> “2000年にXPメーリングリストで提唱（その後、論争）

http://c2.com/cgi/wiki?AcceptanceTest

MF's Bliki - 実例による仕様

“XP/Agile Universe 2002でSbEに心を奪われた

http://capsctrl.que.jp/kdmsnr/wiki/bliki/?SpecificationByExample

（例）年次有給休暇 - 日数

- 使用者は、その雇入れの日から起算して6箇月間継続勤務し全労働日の8割以上出勤した労働者に対して、継続し、又は分割した10労働日の有給休暇を与えなければならない（労働基準法第39条第1項）。
- さらに1年間、8割以上継続出勤するごとに有給休暇は10労働日に加えて勤続2年6箇月目まで1労働日ずつ加算して付与され、勤続3年6箇月目からは2労働日ずつ加算して付与される。勤続6年6箇月経過時には20労働日に達し、以降は1年間の継続勤務ごとに20日を付与すればよい（労働基準法第39条第2項）。

http://ja.wikipedia.org/wiki/年次有給休暇

（例）年次有給休暇 - 日数

表にするとわかりやすい!!

継続勤務年数	0.5年	1.5年	2.5年	3.5年	4.5年	5.5年	6.5年以上
法定最低付与日数	10日	11日	12日	14日	16日	18日	20日
上記を上回る日数の加算は法定外となるので使用者の自由裁量となる。							

※実際には有効期限（2年）があるので面倒だけど。

http://ja.wikipedia.org/wiki/年次有給休暇

INVEST by  Bill Wake

- テスト可能であること
  - 完了や受入が可能であること
- 実例が思い浮かぶこと
  - ストーリーカードの裏側にメモる

ストーリーの裏書こわい!!

『ナニワ金融道 (2)』（青木雄二）
http://www.amazon.co.jp/dp/4062605511

INVEST by  Bill Wake

- テスト可能であること
  - 完了や受入が可能であること
- 実例が思い浮かぶこと
  - ストーリーカードの裏側にメモる
  - 書きすぎて契約みたいになると困る

「受入可能」
なのに
「書きすぎない」
って何なんだよ！

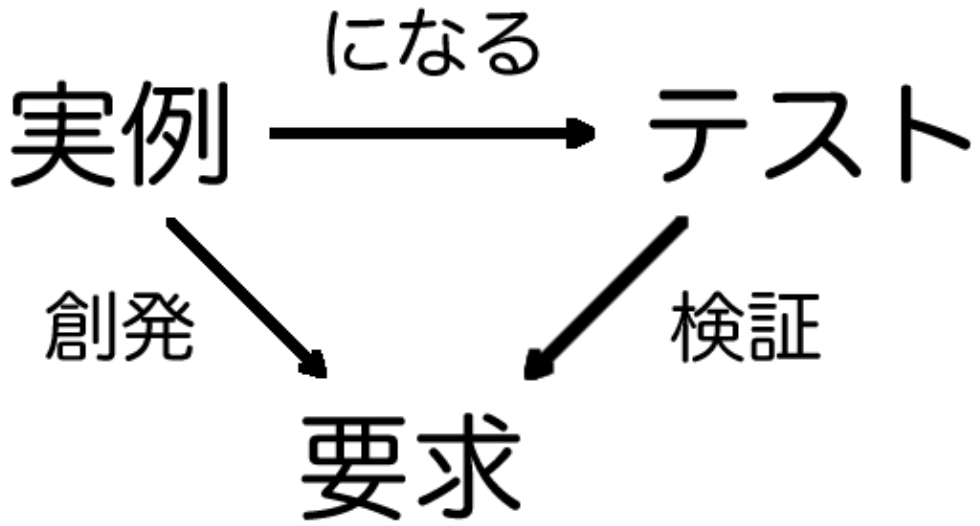
裏書
のヒント

1. チームで一緒に作る

『Wiki Way コラボレーションツールWiki』でチーム萌え

2. 実例（例えば...）から開始

テストは段階的に整理していく



『Bridging the Communication Gap』(Gojko Adzic)

3. ツールの記法を参考にする

いずれもユーザーの視点から（ATDD）

- 表形式
  - FIT/FitNesse, Selenium, Robot Fw
- テキスト形式
  - Exactor, TextTest
- Gherkin（GWT）形式
  - Cucumber, Cuke4Nuke

GWTの日本語化

```
$ cucumber --i18n ja
   | feature                | "フィーチャ", "機能"
   | background             | "背景"
   | scenario               | "シナリオ"
   :
   :
```

- Given ⇒ 前提 ...
- When ⇒ もし ...
- Then ⇒ ならば ...

4. 晴れの日と雨の日のシナリオ

- 晴れの日（正常系）を中心に
- 雨の日（異常系）を追加的に

降水確率はかなり高め!!

（例）ATM利用の晴れの日

顧客として、ATMからお金を引き出したい。

- シナリオ: お金を正常に引き出す
- 前提 口座に10万円入っている
- もし 金額に30,000円と入力する
- ならば 3万円が引き出されていること

ヒント1～4のまとめ

チームで一緒に
実例を使いながら、
ツールの記法を参考に
晴れと雨を意識する。

雨の日のシナリオ（8分）

顧客として、ATMからお金を引き出したい。

- シナリオ: [＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿]
- 前提［＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿］
- もし［＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿］
- ならば［＿＿＿＿＿＿＿＿］いること

グループでいくつか作ってみてください。

ここまでのまとめ

- 最初からテストを目指さない
- 実例を使う（例：10万や3万）
- チームで一緒に考える
- 自動化ツールの記法で考える
- とりあえず晴れの日を考える

75分経過

14:35

残り45分

（参考）GWTも三幕構成である

- 前提 口座に10万円入っている
  ⇒ 状況や設定
- もし 金額に30,000円と入力する
  ⇒ 葛藤や行動
- ならば 3万円が引き出されていること
  ⇒ 理由や結末

第3部 パターン、リーン、ユーザーストーリー

「物語はありません」問題

「仮面ライダー W&ディケイド MOVIE対戦2010」より

物語のない2つの難しさ

- Complicated（入り組んだ）
  - 全体 ＝ 合計(部分)
  - 例：機械の部品
- Complex（複雑な）
  - 全体 ≠ 合計(部分)
  - 例：生命的なもの

Complicated

http://www.amazon.co.jp/dp/B002YK5T9G

Complex

http://www.amazon.co.jp/dp/B000S0HZYG/

Complicated

http://www.amazon.co.jp//dp/B004S8MKW6/

Complex

http://www.amazon.co.jp/dp/B004S8MKJY

※魂を持つ「超ロボット生命体」

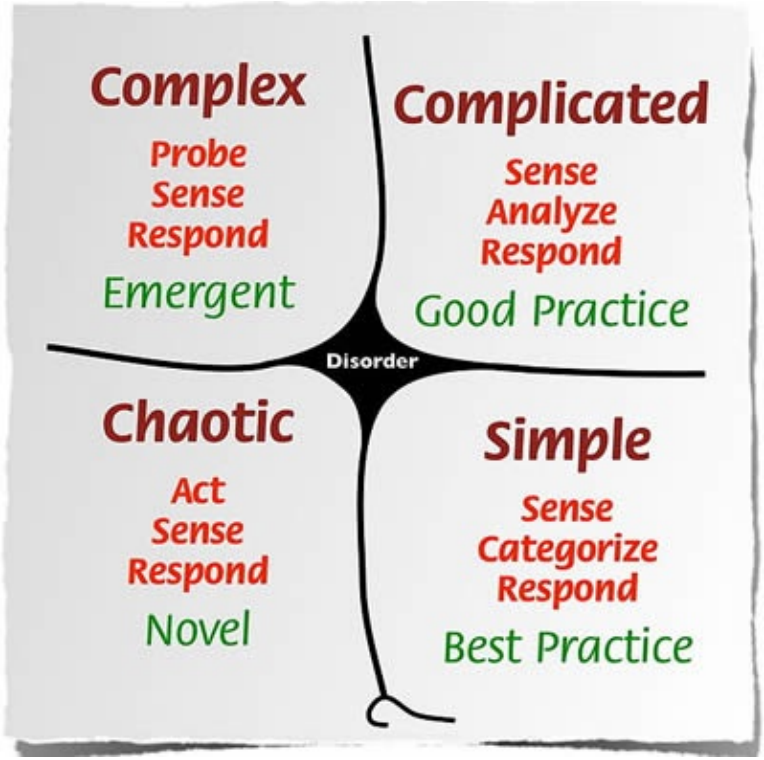
2つの難しさ（3分）

- 身近にある Complex なものと Complicated なものについて、
- グループで話し合ってください。

※ これ自体が「難しい」ワークショップですが＞＜

複数の難しさに対応する

クネビンフレームワーク



http://en.wikipedia.org/wiki/Cynefin

Complexの対応指針

- 何がわからないかわからない
- 探索 → 把握 → 対応
- 環境を整えて実験を繰り返す
- 相互交流とコミュニケーション
- パターンを創発する

『ハーバード・ビジネス・レビュー』2008年3月号

パターンを創発する

Complicatedの対応指針

- 「わからない」と認識している
- **把握** → 分析 → 対応
- **因果関係**を探せば見つかる
- 意思決定に**時間がかかる**

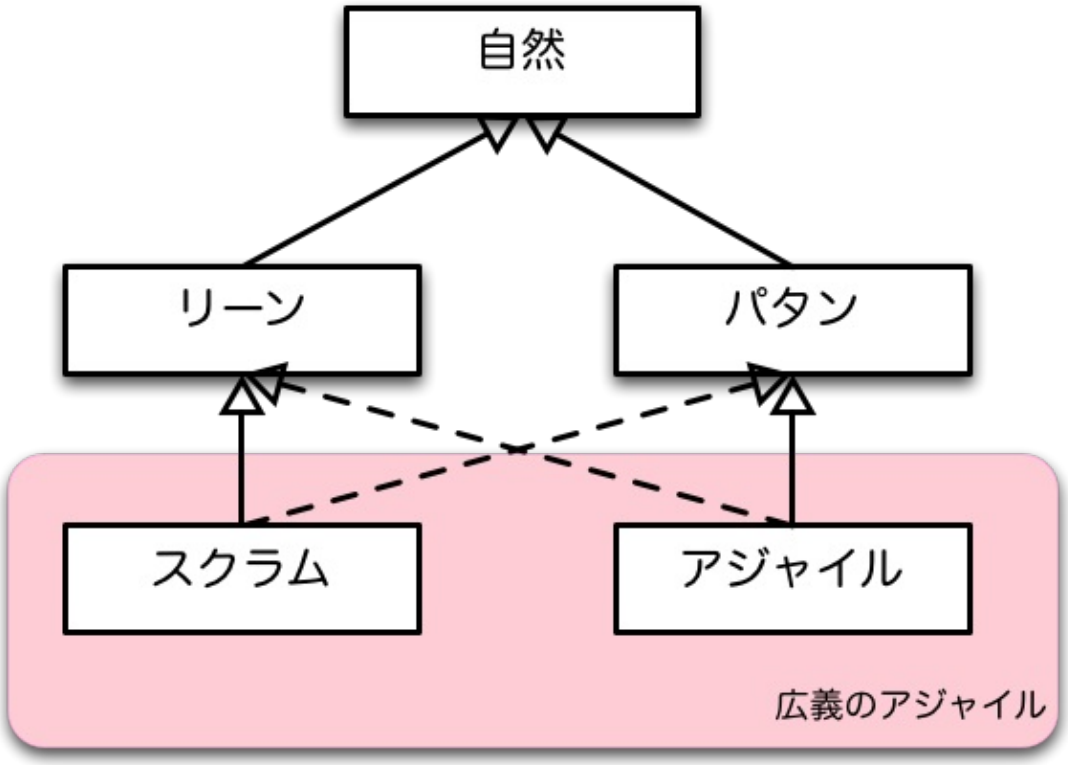
『ハーバード・ビジネス・レビュー』2008年3月号

意思決定に
時間がかかる
↓
探せば見つかる

ザ・トヨタウェイ：原則13

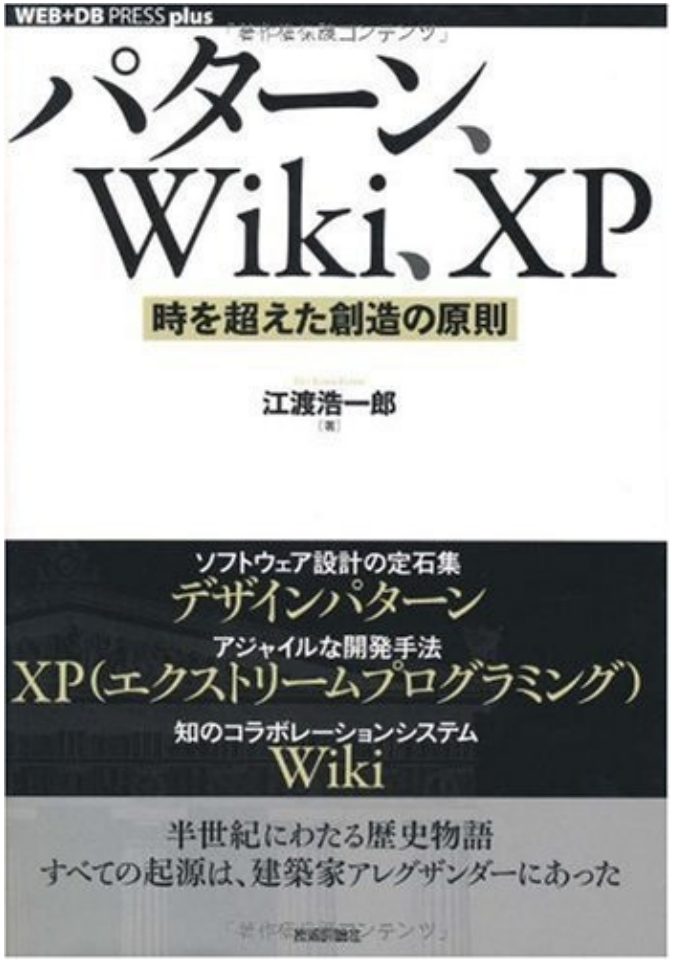
“意思決定はじっくりコンセンサスをつくりながら、あらゆる選択肢を十分に検討するが、実行は素早く行う（根回し）。

パターン&リーンを思い出せ

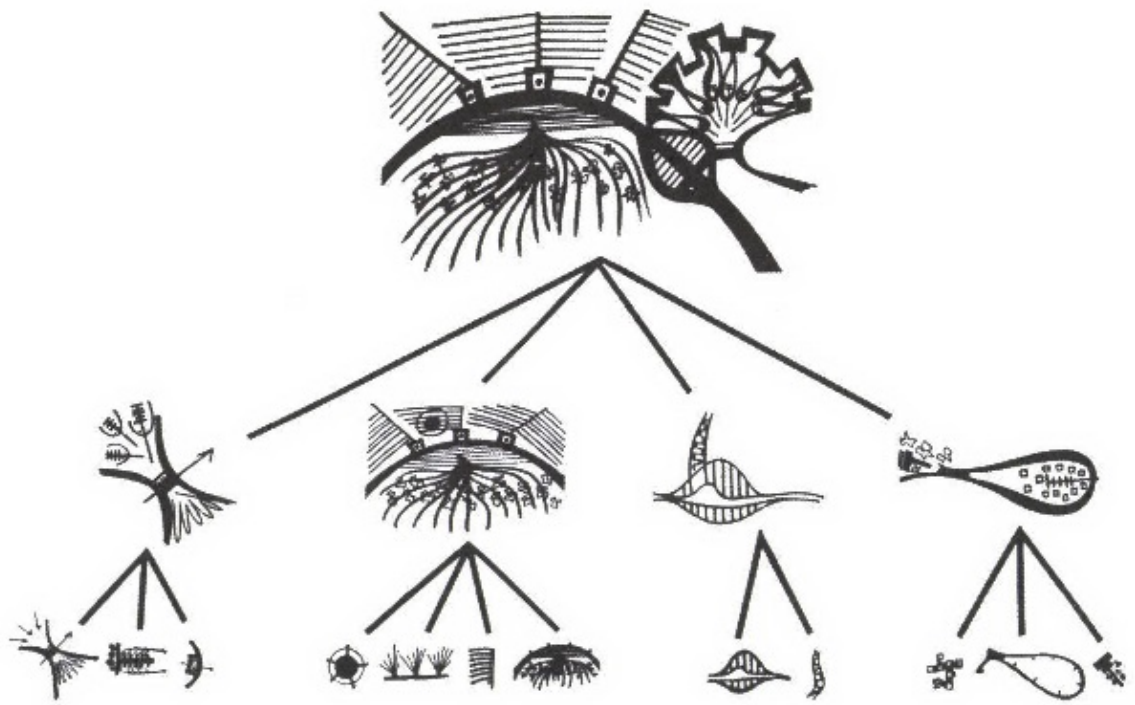


参考：http://objectclub.jp/event/2010alexande

パターン、Wiki、XP


WEB+DB PRESS plus
パターン、Wiki、XP
時を超えた創造の原則
江渡浩一郎
ソフトウェア設計の定石集
デザインパターン
アジャイルな開発手法
XP(エクストリームプログラミング)
知のコラボレーションシステム
Wiki
半世紀にわたる歴史物語
すべての起源は、建築家アレグザンダーにあった

都市はツリーではない

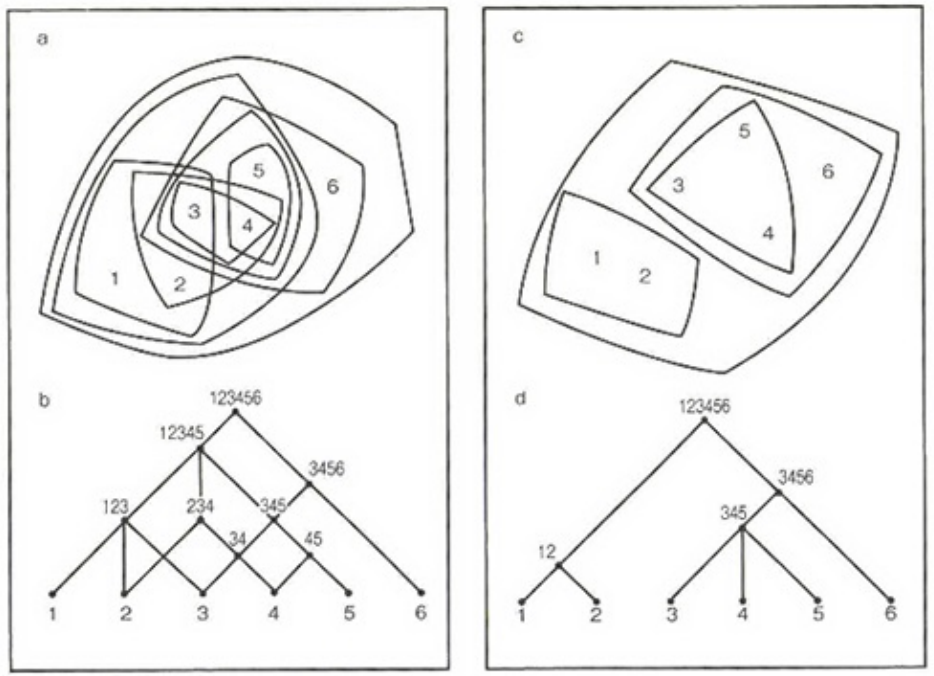


❖図1.4　インドの村の分析を1つにまとめたダイアグラム

※C. アレグザンダー 著／稲葉武司 訳『形の合成に関するノート』鹿島出版会、1978、p.138

『パターン、Wiki、XP』（江渡浩一郎）p.17より

都市はツリーではない



❖図1.6　セミラティス構造(a、b)とツリー構造(c、d)

※Christopher Alexander, "A city is not a tree", Architectural Forum, Vol.122, 1965(鹿崎新著『建築の解体――一九六八年の建築情況』鹿島出版会、1997、p.178より)

『パターン、Wiki、XP』（江渡浩一郎）p.21より

ストーリーはツリーではない

※写真はイメージです

アレグザンダーの6つの原則

1. 有機的秩序の原則
2. 参加の原則
3. 漸進的成長の原則
4. パターンの原則
5. 診断の原則
6. 調整の原則

※青色はユーザーストーリーと関係のある原則
参考：『パターン、Wiki、XP』（江渡浩一郎）

17章. EPISODES

- 製品始動計画パターン
- 市場のおさらいパターン
- **暗黙の要求事項パターン**
  - 翻訳が悪い（Implied：**暗示的**）
- 作業待ち行列パターン

ユーザーストーリーは
パターンである

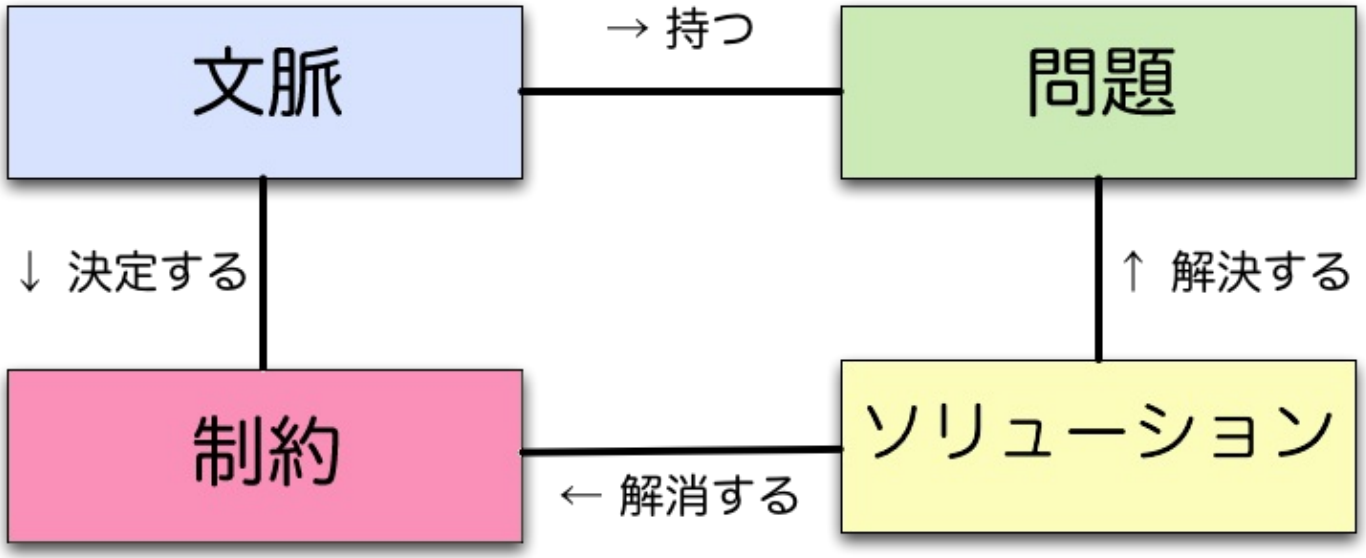
パターンであれば、
パターンのように生成
……できるはず。

パターン by C. アレグザンダー

“ある「文脈」で繰り返し起きる「問題」を「解決」する方法。その方法にはいくつかの「制約」が課せられているかもしれない。—  結城浩

http://www.hyuki.com/dig/patlang.html

パターンの構造



参考：A Pattern Language for Pattern Writing（Gerard Meszaros, Jim Doble）
参考：パターンランゲージ2010 Class #9 Pattern Writing, part 2（井庭 崇）

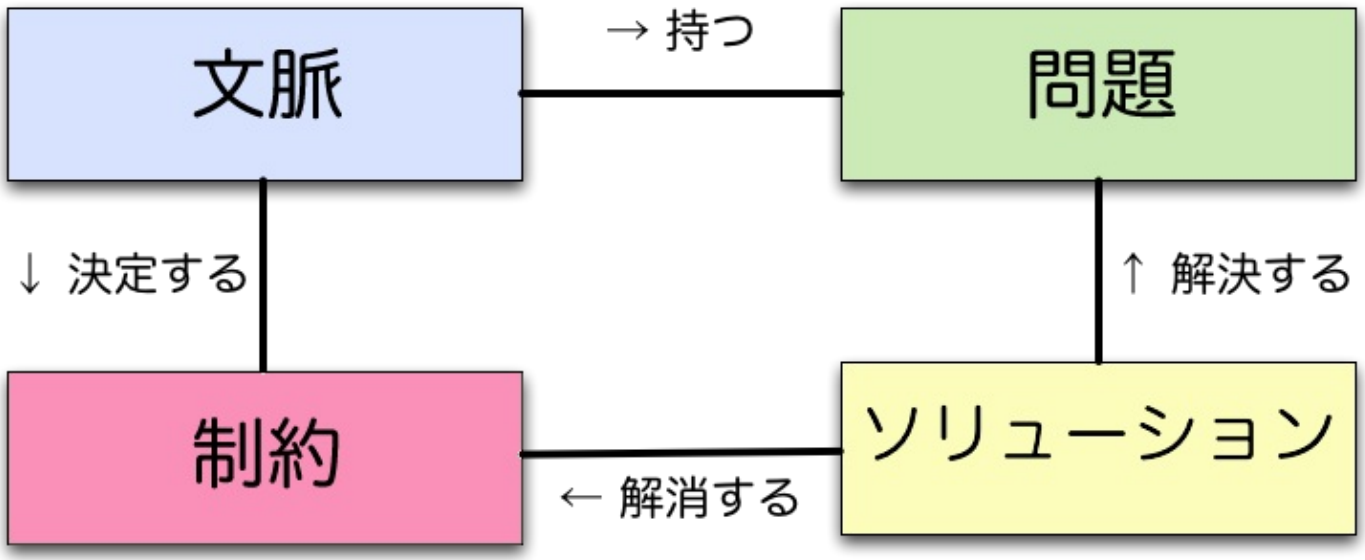
パターンのように生成する

1. うまくいっている事例から
2. うまくいっていない事例から
3. 理論的な側面から

参考：パターンランゲージ2010
Class #5 Pattern Mining（井庭 崇）
http://gc.sfc.keio.ac.jp/cgi/class/class_top.cgi?2010_25136

パターンの構造と生成

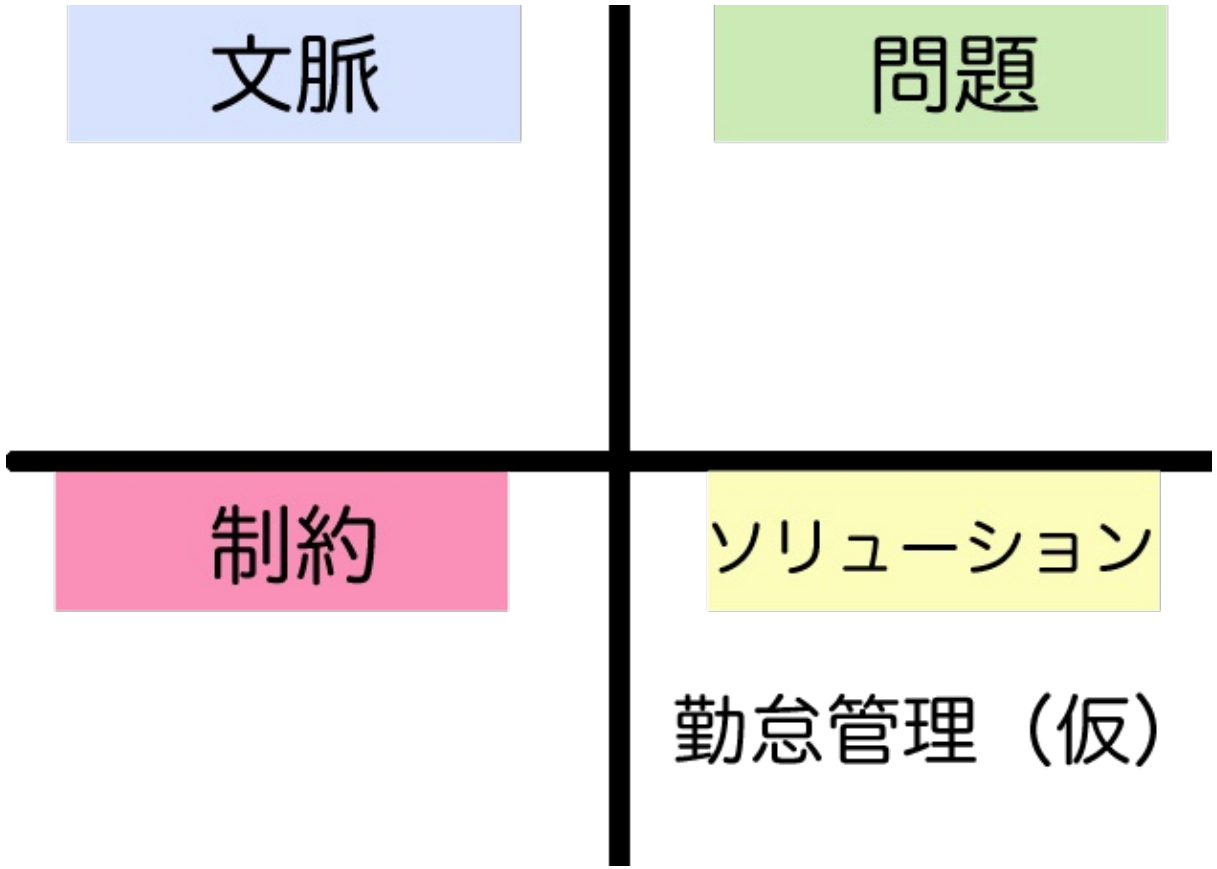
- うまくいっている事例 ⇒ 「ソリューション」起点
- うまくいっていない事例 ⇒ 「問題」起点
- 理論的な側面 ⇒ 「文脈」「制約」起点



参考：A Pattern Language for Pattern Writing（Gerard Meszaros, Jim Doble）
参考：パターンランゲージ2010 Class #9 Pattern Writing, part 2（井庭 崇）

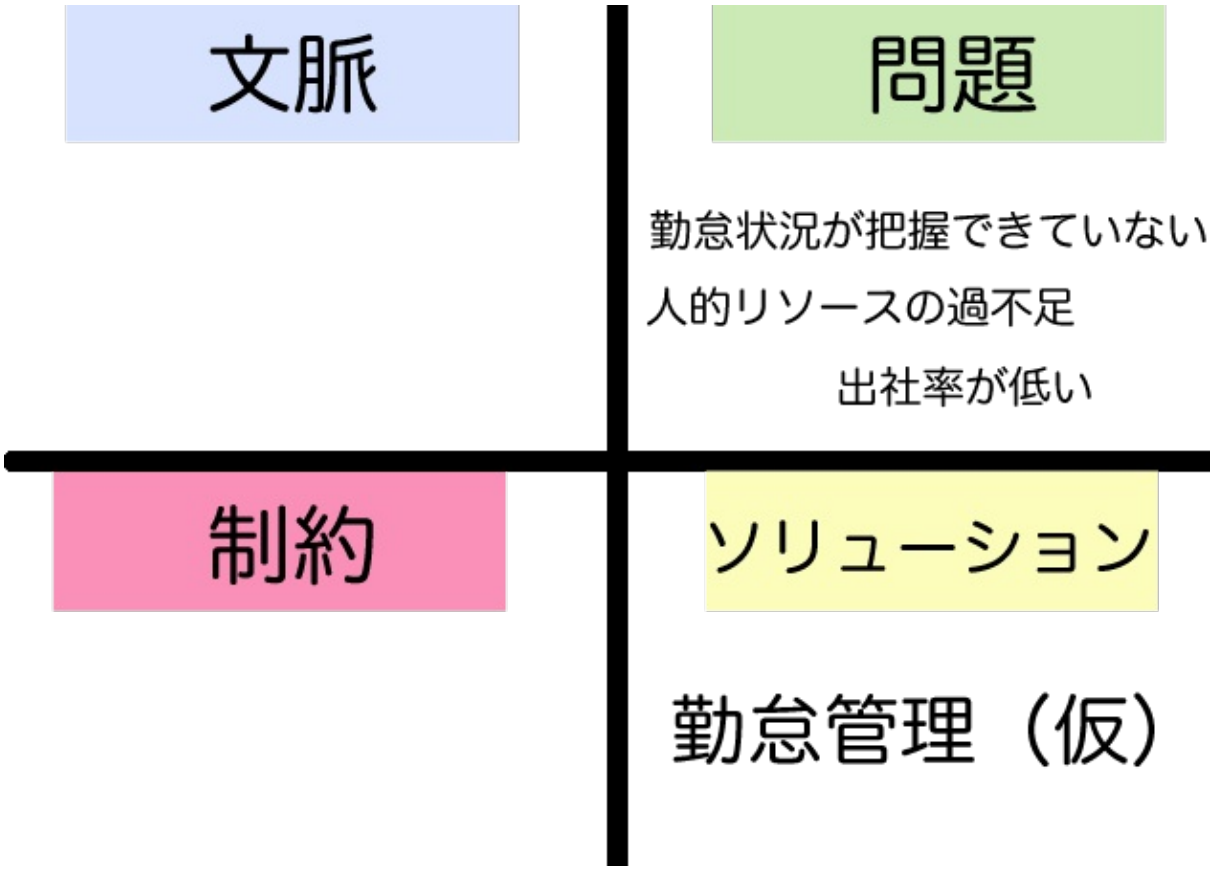
（例）勤怠管理を導入したい

テーマをソリューションに仮置きして開始



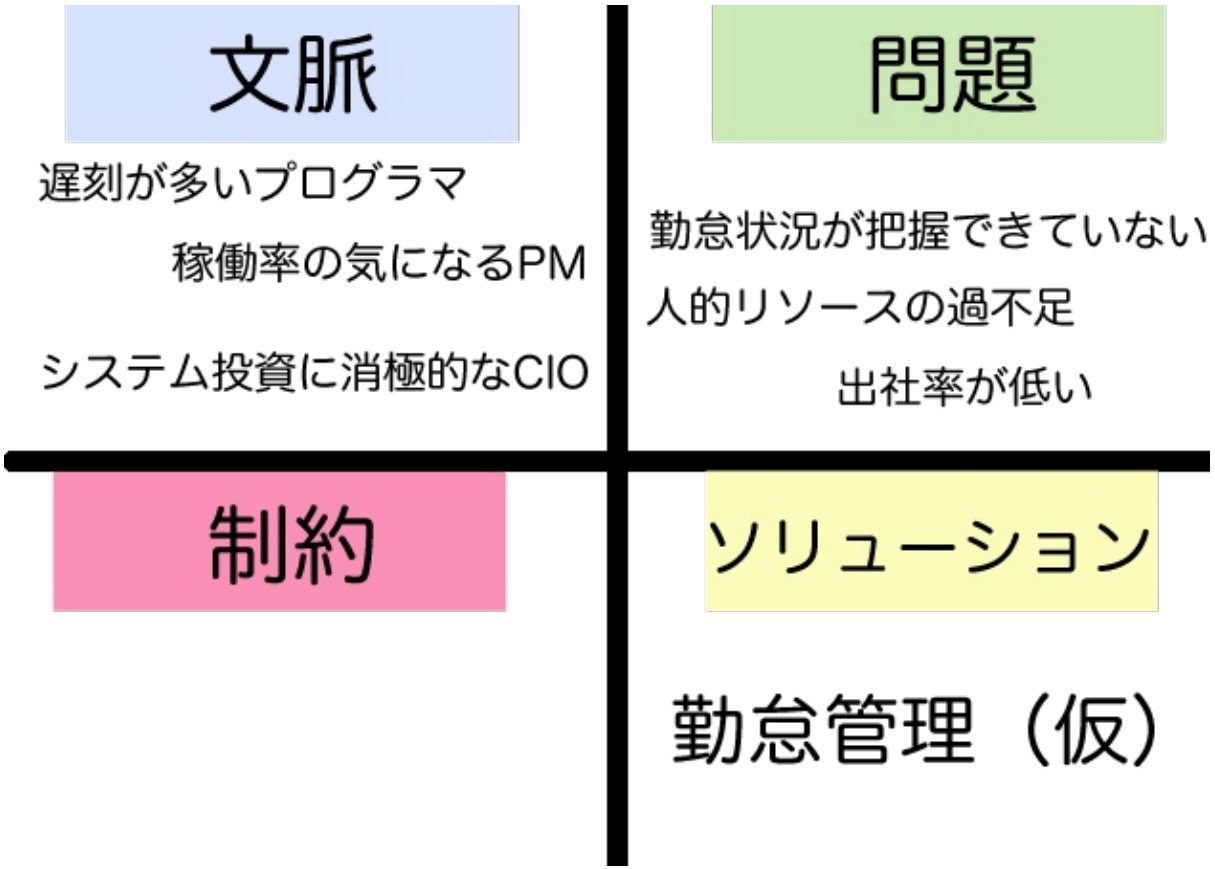
（例）勤怠管理を導入したい

それが解決しそうな問題を列挙



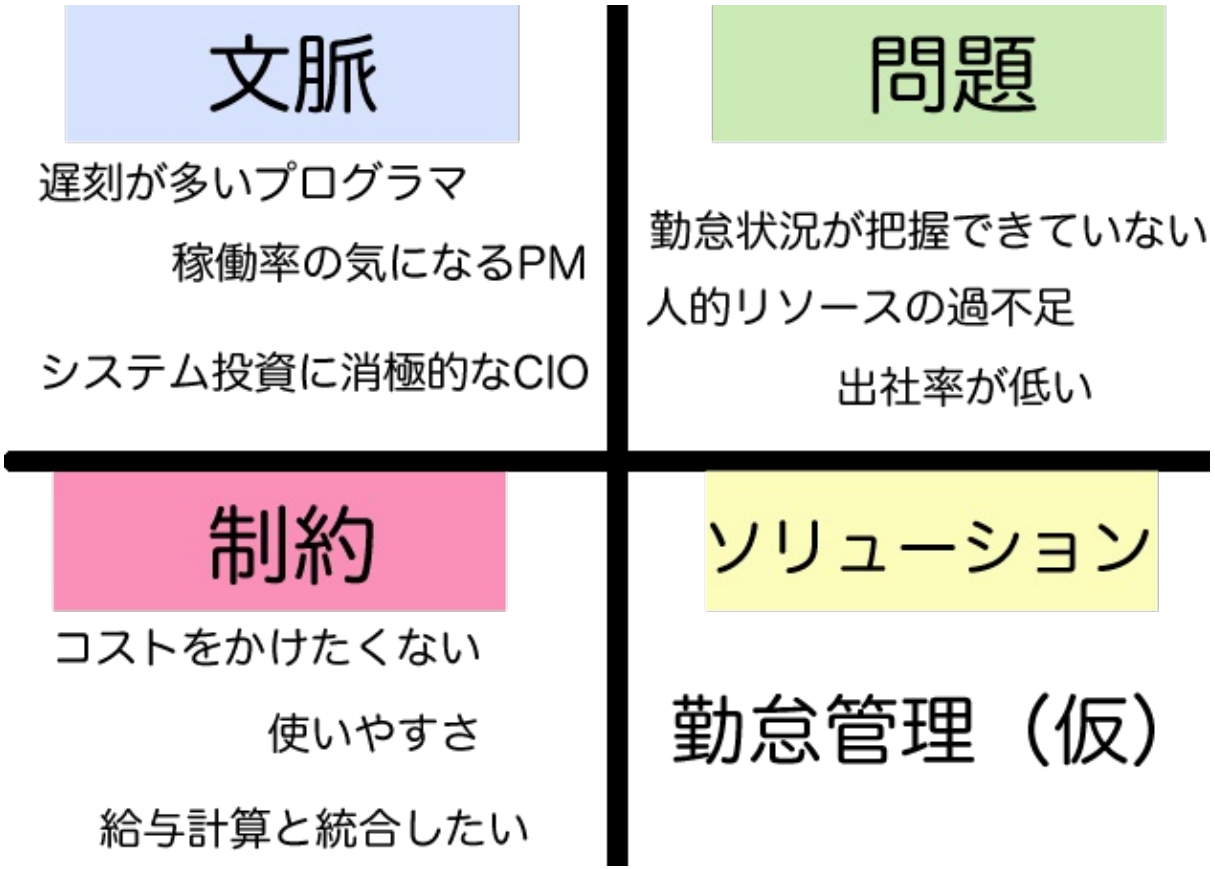
（例）勤怠管理を導入したい

問題を持っていそうな文脈を想像



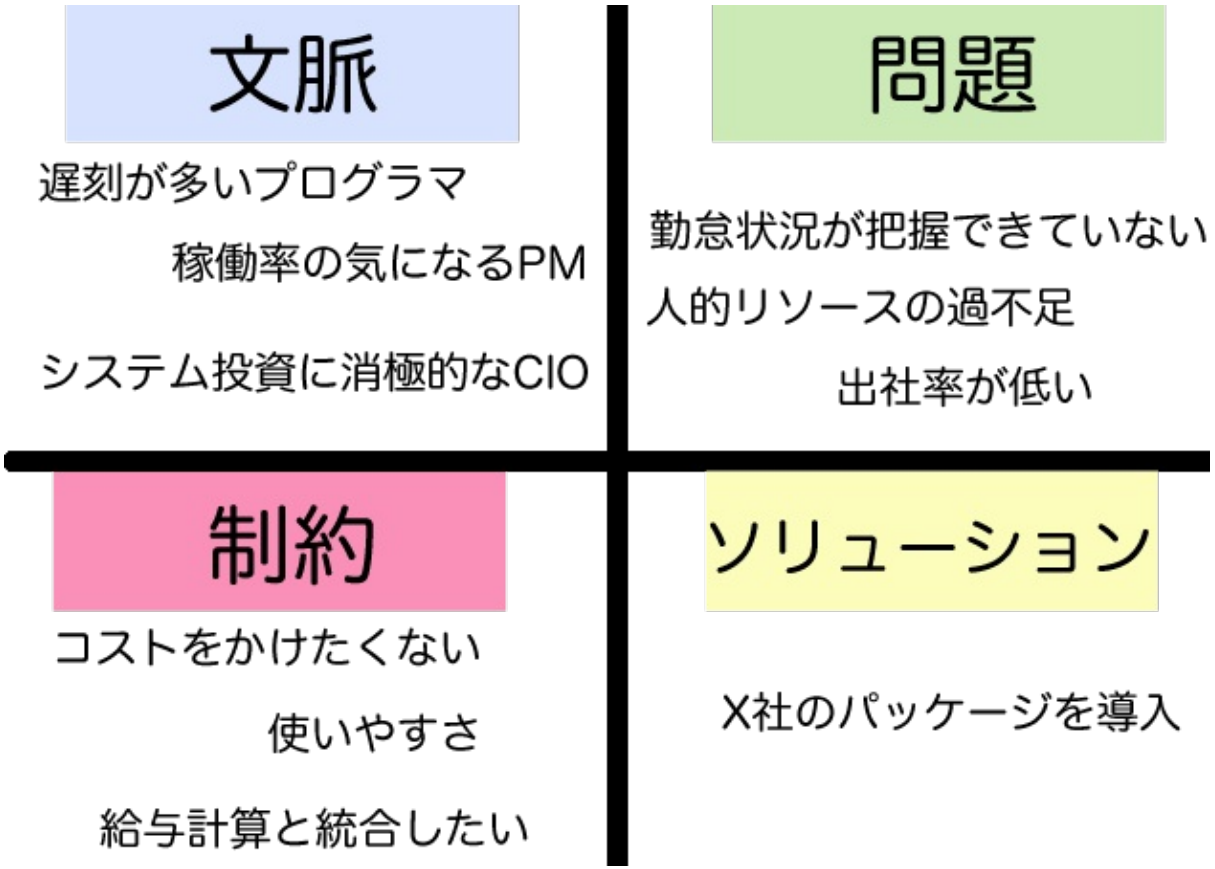
（例）勤怠管理を導入したい

文脈は同時に制約を決める



（例）勤怠管理を導入したい

ソリューションを見直す



（例）勤怠管理を導入したい

ぐるぐる回していく（順・逆方向）

（参考）問題と制約の違い

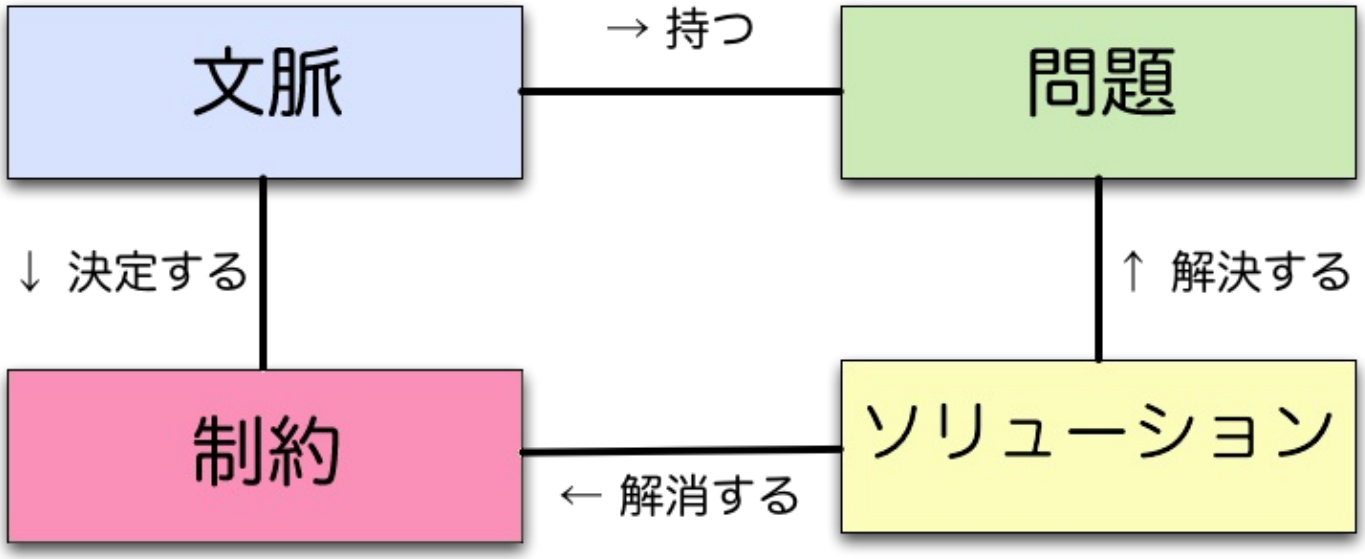
切るなら「問題」
残すなら「制約」

http://www.flickr.com/photos/mrimperial/135375001/

在宅勤務を導入したい

4色カードで書き出してみてください（8分）

- うまくいっている事例 ⇒ 「ソリューション」起点
- うまくいっていない事例 ⇒ 「問題」起点
- 理論的な側面 ⇒ 「文脈」「制約」起点



在宅勤務を導入したい(3分)

「文脈」「問題」「制約」から

3枚1組の組み合わせを

いくつか作ってみてください。

※ 要素は重複しても構いません。

パターンはボトムアップ手法

> “ボトムアップデザイン は、ソフトウェアがどんどん複雑化していくなかで一層重要になってきている。
> ― 『On Lisp』（ポール・グレアム）

100分経過

15:00

残り20分

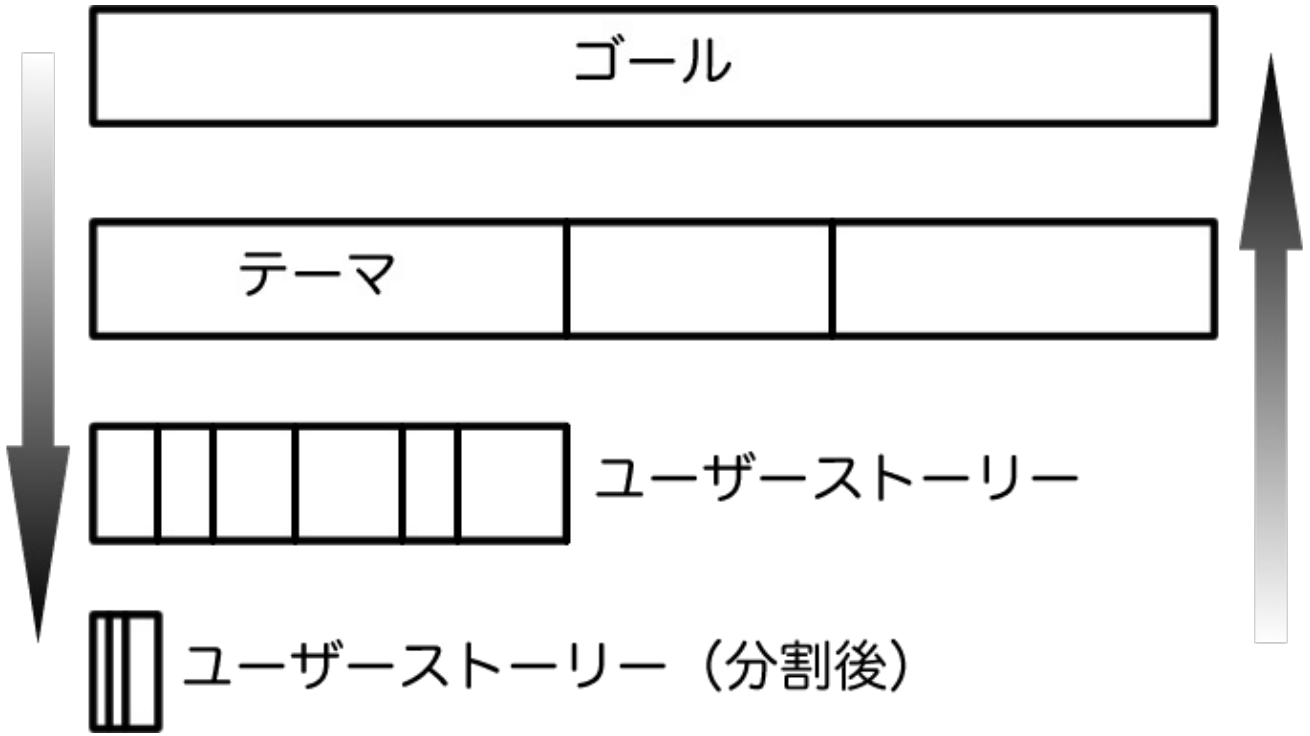
リーンの概念

- **自働化** - 品質の作り込み
  - TDD・ATDD・CI
  - 見える化・職能横断型組織
- **JIT** - ムダ・ムラ・ムリの排除
  - PBI・PO・スプリント
  - プルベース・作業の平準化

http://www.toyota.co.jp
/jpn/company/vision/production_system/

Complicatedなら分割できる

5つのなぜで発掘（トップダウン）



（参考）ストーリーの分割

- データ境界
- 操作の境界
- 横断的な関心事
- パフォーマンス制約
- 優先度

『アジャイルな見積りと計画づくり』12章

(参考) ストーリータイプで分割

1つのストーリーの4つの側面

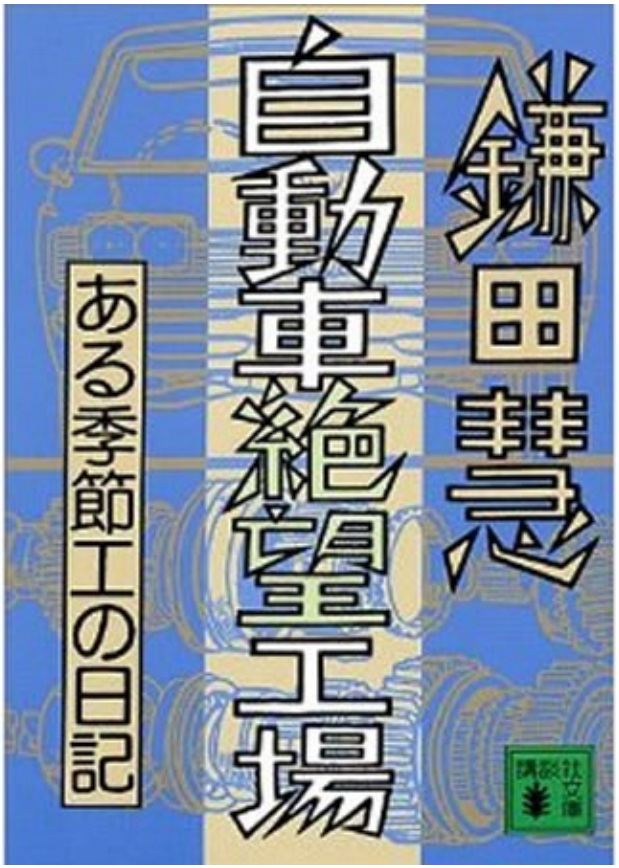
- 基本機能
- 派生機能
- ビジネスルール
- ユーザビリティ

http://storyotypespaper.gerardmeszaros.com/

パターン&リーンの使い分け

- 「かたい」ところはリーン
  - 把握 → 分析 → 対応
  - 時間をかけて意思決定
  - ソフトウェア要求・ドメインモデル・アーキテクチャ
- 「やわらかい」ところはパターン
  - 探索 → 把握 → 対応
  - コミュニケーション・実験・創発
  - システム要求・UI・イノベーション

使い方を間違えてはいけない



『ゆとりの法則』（トム・デマルコ）の方がいいなぁ

ここまでのまとめ

- 物語がないなら2つの難しさがある
- ストーリーの源流はパターンだ
  - パターンの構造を参考にして、ストーリーをボトムアップで生成する
- アジャイルはリーンでもある
  - 入り組んでいれば、ツリーのようにトップダウンで分割できる

105分経過

15:05

残り15分

参考書


Succeeding with Agile
Agile Testing
Specification by Example
Agile Software Requirements
Dean Leffingwell
鎌田君急
自動車絶望工場
ある季節工の日記
Lean Architecture
Agile Software Development SECOND EDITION
The Cooperative Game
パターン、Wiki、XP
時を超えた創造の原則
江渡浩一郎
プログラムデザインのための
パターン言語
アジャイルな
見積りと計画づくり
Bridging the Communication Gap
User Stories Applied
For Agile Software Development
Growing Object-Oriented Software, Guided by Tests
デザインパターン
XP(エクストリームプログラミング)
Wiki
半世紀にわたる歴史物語
すべての起源は、建築家アレグザンダーにあった
映画を書くために
あなたがしなくては
ならないこと
Fit for Developing Software
Framework for Integrated Tests
Rick Mugridge
Ward Cunningham
The Cucumber Book
The Clean Coder
A Code of Conduct for Professional Programmers
TEST DRIVEN
Practical TDD and Acceptance TDD for Java Developers
manning
せかい いち
おいしいスープ
Managing Software Requirements
Second Edition
A Use Case Approach
Dean Leffingwell
Don Widrig
extreme Programming explained
Embrace Change
Kent Beck
Foreword by Erich Gamma

今日のまとめ

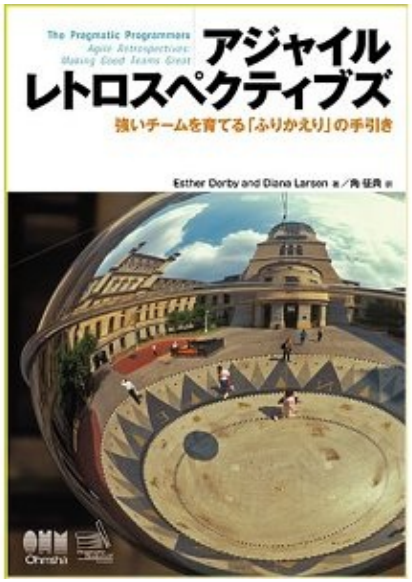
- ビギンズナイト・スクラムガイド
- 石・三幕構成・続きは質問
- 条件はみんなで実例とツールを
- 2つの難しさ・パターン・リーン

ふりかえり（2分）

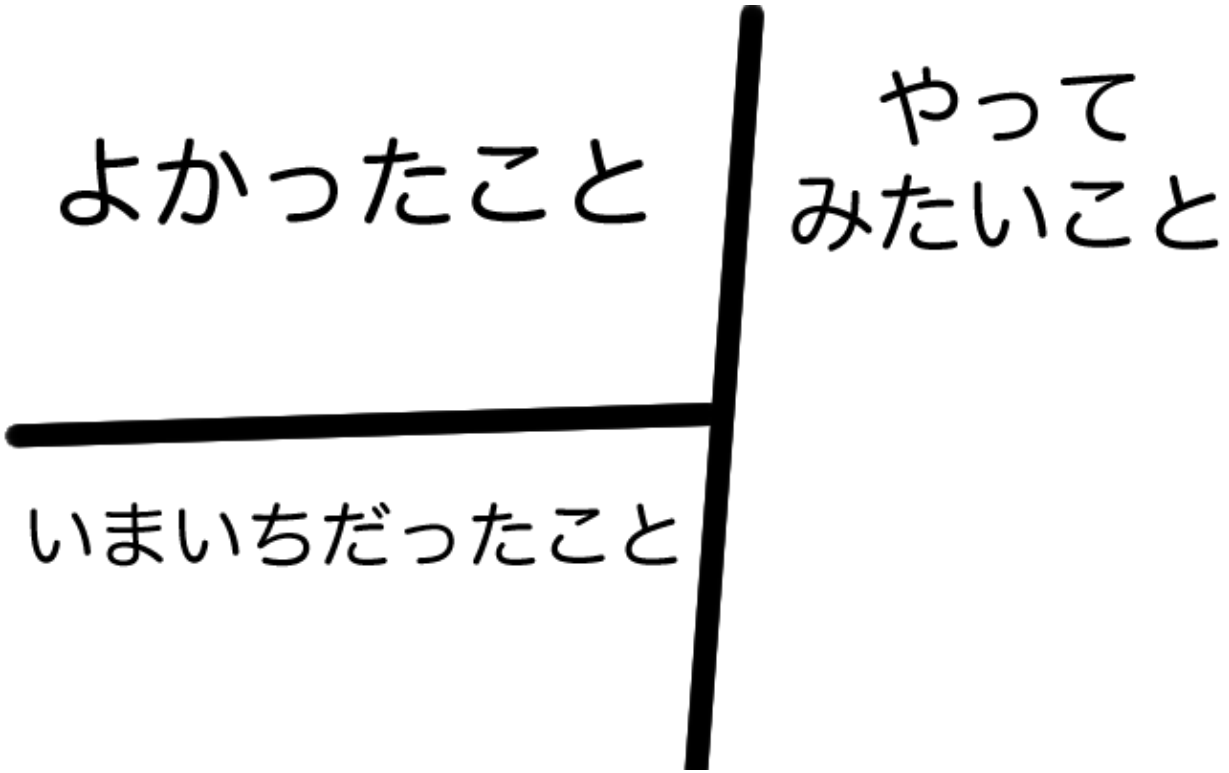
今日の**ワークショップ**を受けてみて、

現場に戻ってやってみたいことを

三幕構成で書いてみてください。



みんなでふりかえり（5分）

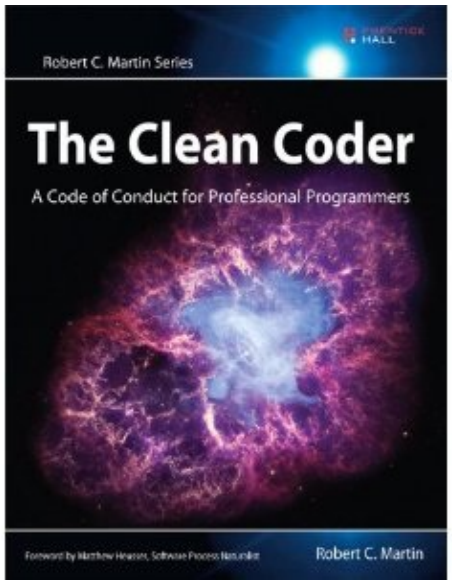


112分経過

15:12

残り8分

何か質問はありますか？（8分）



kado.masanori@waicrew.com
twitter: @kdmsnr

http://www.slideshare.net/kdmsnr