3. 体験チーム開発

3.1 アプリケーションのアイデアを決める

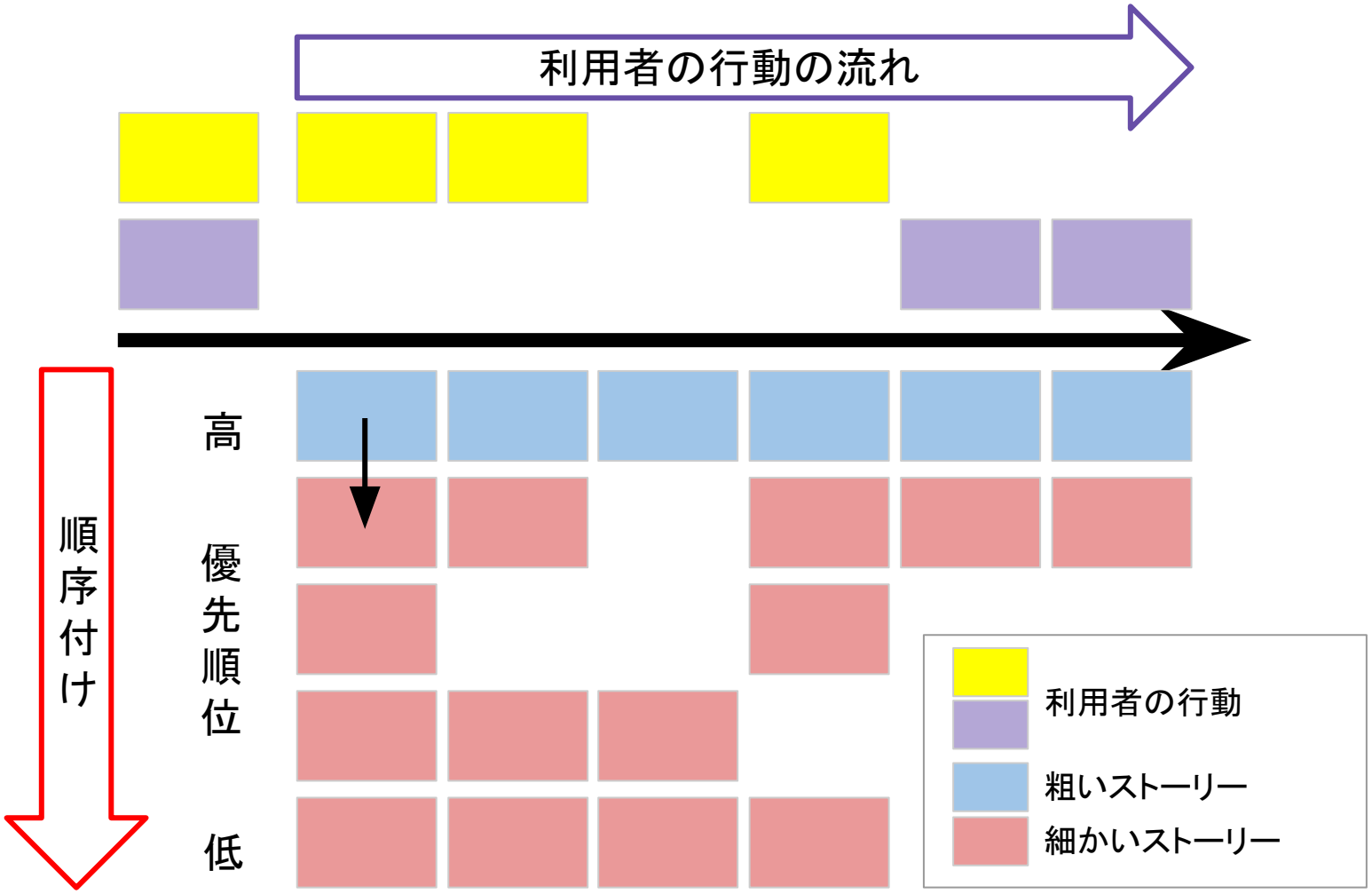
3.1.1 555（Triple Nickels）のやり方

1. 5人程度のグループを作る
2. 各自5分間でブレインストーミングし、
   アイデアを紙に書き出す
3. 隣の人に紙を渡し、隣の人は書かれたアイデアに
   関連する新しいアイデアを追記する
4. 5分経ったら、また隣の人に渡す

（紙が最初に書いた人のところに戻ってくるまで
繰り返す）

3.1.2 インセプションデッキで プロダクトの特徴を明確にする

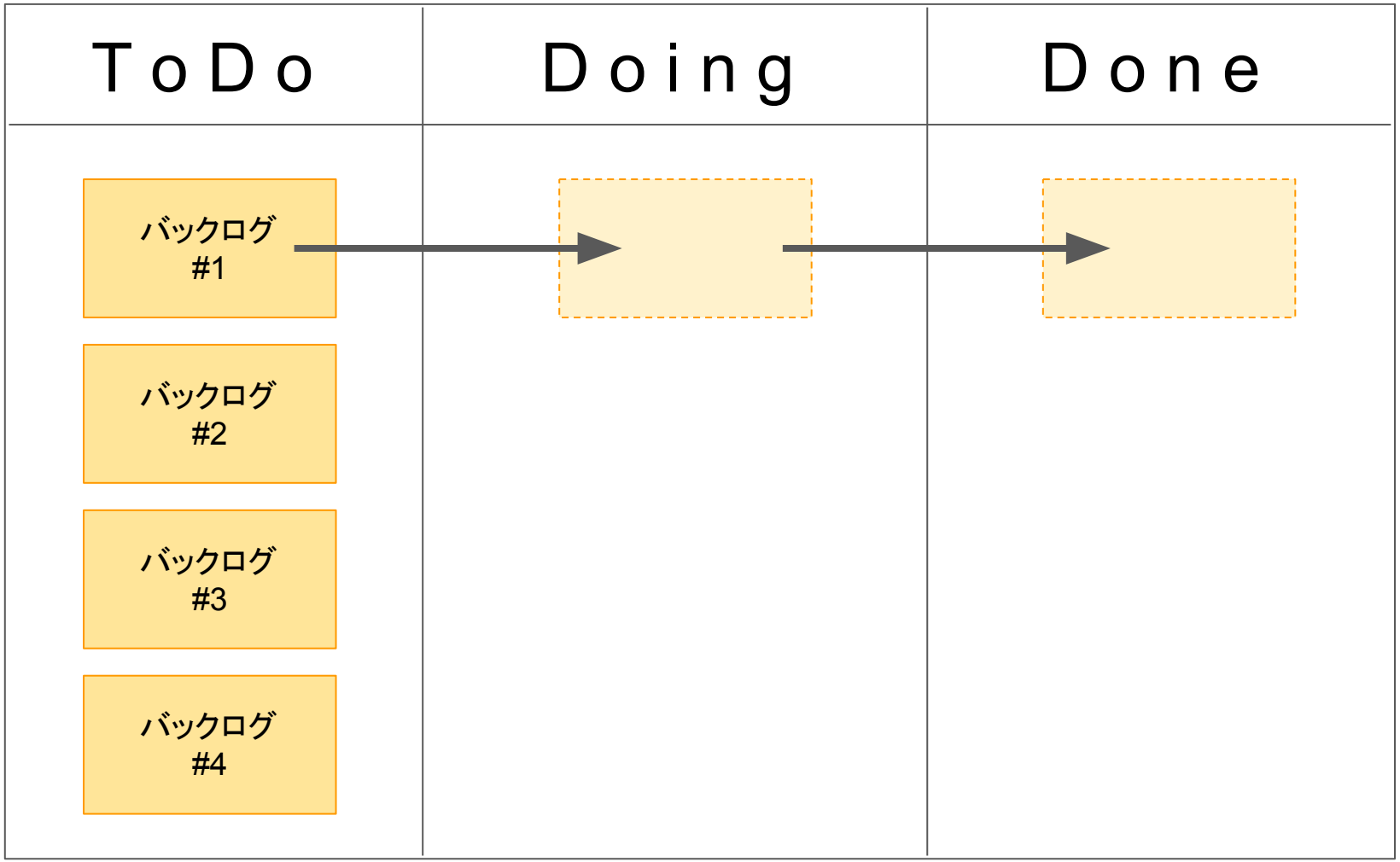
3.1.3 ユーザーストーリーマッピングで プロダクトの機能を洗い出す



3.1.4 バックログに優先順位をつけ、プランニングポーカーで見積もる

- みんなで見積もる
- 認識を合わせる
- 時間をかけ過ぎない
- 大きな数字の
  小さな誤差は気にしない

3.1.5 カンバンを作る



3.2 チーム開発を体験する

【実践すべきプラクティス】

・テストコード（テスト自動化）

・ペアプログラミング
  ・チーム内でコマ毎にローテーションを行う

・コマ毎にカンバンを更新する

・リファクタリング

・GitHub Flowの流れでレビューを実施する

3.2.1 ペアプログラミング

ふたりのプログラマがドライバとナビゲータという2つのロールにわかれて
１台のコンピュータに向かい、コミュニケーションを取りながら
設計・アルゴリズム・コード・テストについて継続的に共同作業します。
品質向上、時間短縮、学習など様々な効果が得られます。

ドライバ　：　コンピュータへの入力や設計の書き下ろしをします。
ナビゲータ：　ドライバの作業を監視し、構文エラー、タイプミス、
　　間違ったメソッドの呼び出しや、実装コードが必要とする
　　機能を満たさないような場合に指摘します。
　　ナビゲータは戦略的に長期的視点から考えます。

3.2.2 Github flowとは

1. 新しい何かに取り組む際はブランチを作成する
2. 作成したブランチで作業し、定期的にサーバーにも作業内容をPushする
3. フィードバックや助言が欲しい時、ブランチをマージしたい場合は、プルリクエストを作成する
4. 他の誰かがプルリクエストをもとにレビューする
5. 修正点があれば修正する
6. 問題なければマージする

3.2.3 チーム開発を体験する

(1) ペアプログラミングを行いながら
本研修で学習したActionCableを使って
チャット機能を実装してみましょう。

(2) 完成したと思えたら、プルリクエストを出し、
他のチームメンバーにレビューしてもらいましょう

(3) 本研修で学習したGemを使って
チャットアプリにログイン機能を実装してみましょう

(4) 完成したと思えたら、プルリクエストを出し、
他のチームメンバーにレビューしてもらいましょう

(5) 開発したアプリケーションを
Herokuにデプロイしてみましょう

(6) スプリントレビューを実施しましょう

・完了したストーリーのデモンストレーションを行い、
顧客役からフィードバックをもらう。

・KPT法でふりかえりを行う

・全体のコードレビューを行い、レビュー指摘対応、
リファクタリングが必要な部分を残り時間で行う。