Scrumメモ

参照「もし女子高生がRedmineで「スクラム」開発をしたら」

＜<http://www.atmarkit.co.jp/fjava/index/index_scrum.html>＞

特徴

* アジャイルプロセスの1つで、高いビジネス価値をより早期に顧客に提供することが可能
* 動作するソフトウェアを速やかに繰り返し確認していく（2週間～1カ月周期で）
* 顧客は要件の優先順位をつける。チームは優先度の高い機能を顧客に納める最良の方法を自分たちで決定
* 2週間～1カ月ごとに動作するソフトウェアをみることができ、そのままリリースするか、別のスプリントで機能拡張するかを決めることができる

------------------------------------------準備------------------------------------------

要件定義

* プロダクトバックログ　＝　要件定義をまとめたもの
  + ストーリー
    - ユーザ視点
    - 誰が何をするか具体的に書く
  + フィーチャー
  + 性能
  + 機能
* プロダクトバックログは基本的にストーリーが多い

優先度

* 優先順位をつける
* ストーリーの重要度　＝　ビジネス価値
  + 「顧客」が欲しい機能ではなく「エンドユーザ」が欲しい機能に高い優先度がつく
  + 無駄な機能の作り込みが防げる
* 開発の生産性や品質を左右するような項目も優先度を高くしたほうがよい
  + 最終的に「ユーザが満足する品質の製品をつくる」という部分につながる

見積もり

* やり方はいろいろある
* 一番簡単なストーリーの難易度ポイントを１として基準にする
  + ５倍難しいストーリーにはポイント５など
  + 「人月」単位がもつ不確定特性への対策となるため
  + チーム作業としてのより正確な見積もりを行うため
  + フィボナッチ数列でポイントをつけると良い
    - 小さいポイントは正確に見積もりやすい
    - 大きいポイントほど曖昧な見積もりになりがち
    - 上述より小さいポイントは細かく、大きいポイントは荒く設定したほうが良い＝＞フィボナッチ数列
* 見積もりは最初に全部やる必要はない
  + 優先順位が高いものから、現行スプリントで取り組むストーリー＋αの見積もりを行う
* 顧客（Scrumではプロダクトオーナーと呼ぶ）と相談して決める

スプリント設定

* スプリント　＝　作業時間の単位（タイムボックス）
  + 基本は２週間〜１ヶ月周期
  + イテレーション
* スプリントごとに動くソフトウェアをつくる
  + 実際に使ってみて仕様漏れや新たな改善点を洗い出す
* まずスプリントの長さを決める
  + 何度か開発を行う内にチームにとって丁度良いスプリント期間を探す

スプリント計画

* 優先順位の高いストーリーから順にスプリント内でできそうな数だけ選ぶ。
* １スプリントで消化できるポイント数　＝　ベロシティ
  + 注意：チームメンバーが変わるとベロシティが狂うので、見積もりの精度が一時的に低下する
  + 固定メンバーでスプリントを何回も回すことが見積もりの精度を上げる
* スプリント計画は臨機応変に
  + 作業量が半端なら、ストーリーを詳細化または他ストーリーと作業順を入れ替えるなど

スプリントバックロッグ

* スプリントバックログ　＝　タスク
  + ストーリーを細かい作業単位に分割したもの
* ストーリーを分解してスプリントバックログ（タスク）とする
* タスクごとにどのくらいの時間がかかるか見積もる
  + 顧客（プロダクトオーナーとの相談は不要）

タスク

次を参考に

「スプリントのタスクに関するTips２９個」

＜<http://www.ryuzee.com/contents/blog/3959>＞

------------------------------------------実践------------------------------------------

デイリースクラム

* 毎回の開発の始まりに行うミーティング
  + 前回行った作業状況の確認
  + 今日何をやるのかの確認
  + 作業を進める上での問題・課題

進捗の確認・管理

* タスクボードやバーンダウンチャートを利用
* かんばん　＝　タスクボード
  + タスクの状態を”To do”, “Doing”, “Done”のステータスに分けてボードに貼る

スプリントレビュー・振り返り

* みんなで動いたものを確認する
  + 顧客がいれば顧客に実際に触ってもらう
* 仕様変更があればプロダクトバックログに追加
  + 優先順位の確認等も必要
* KPT法などを用いてもよい

------------------------------------------Redmine概要--------------------------------

* バックログ管理画面
  + ストーリーをワンクリックで作成できる
  + ドラッグ＆ドロップでスプリントに割り当てられる
* かんばん画面（スクラムタスクボード）
  + ストーリーごとの進捗状況と残り時間を一目で確認できる
  + 担当者ごとに色を割り当てられる（未割り当てはグレー）
  + タスクの残り時間と作業時間を入力すれば、自動的にバーンダウンチャートを作成できる

--------------------------------------Scrum - Redmine--------------------------------

ストーリー（バックログ）の作成

* バックログ画面で作成できる
  + 左：各スプリントのストーリー
  + 右：スプリントに未割り当てのストーリー
* ストーリーの作成
  + プロダクトバックログの左上をクリック
  + 新しいストーリーを選択
  + ストーリーの概要を入力
  + （決まっているなら）ストーリーポイントの入力

受け入れ条件の記入

* 見積もりを正確に行うために用いられる
* チケットの編集画面→鉛筆マーク（説明）部に書き込める
* ファイルとしても添付可
* （ユーザ目線の）具体的な機能を書く（特にフォーマットはなし）

優先順位の並び替え

* 前提：プロダクトバックログが完成している
* ドラッグ＆ドロップで優先順位の順に並び変える

※ここまでは顧客（プロダクトオーナー）と一緒に行った方が良い

---------------------------

スプリント作成

* プロダクトバックログのメニュー→新しいスプリント
  + スプリント名、開始日、終了日を入力
  + （任意）スプリント名はNo.＋到達目標点など、工夫する

ストーリーのポイントの設定

* すべて決める必要はない
  + 優先順位の高い順から、次のスプリントでやりそうなストーリー＋αにのみつけておけばいい
* ストーリー編集画面で設定できる
* スプリントの右上にストーリーポイントの合計が表示される
  + 前回のスプリントのベロシティを考慮してストーリーの割り当てを行うと良い

タスク（スプリントバックログ）の作成

* かんばん画面から行う（スプリント左上メニュー→かんばん）
  + スプリントのストーリー上の”+”をクリック
  + 題名と残り時間を入力
  + 残り時間はタスクの右下に表示
* スプリント計画段階では担当者を決めない（ウォーターフォールとは違って）

デイリースクラム

* かんばんを見ながら行う
* 前回行ったことの確認
  + タスクのステータスの変更
* 今日やることの確認
  + 残時間の確認、タスクのアサイン

スプリント障害事項

* 障害事項を記述する
* ブロック欄に妨害されるストーリーやタスクを記入