

# チーム開発実践入門

2.1-2.3

kitoh

## 2.1-2.3で取り扱われている内容

1. 重要なメールが多すぎて優先順位が決められない(障害管理)
2. 検証用環境がない(検証環境)
3. 別名フォルダでブランチを管理している(ブランチ管理)
4. データベースの再作成が困難(データベース)

今まで携わってきたプロジェクトを振り返ってみた。

- K●B●:無線モジュール開発 (組み込み)
- コ○ボ:サーバーモジュール開発 (クラウド)

内容	K●B●	コ○ボ
1.障害管理	NotesDBで管理	redmineのチケットで管理
2.検証環境	通信相手の用意が困難	AWS上にデプロイ
3.ブランチ管理	プロジェクト全体はgitで管理 チームではフォルダで管理	branchで管理 開発するときはfeatureブランチを作る。
4.データベース	----- ----- -----	masterをpullすると、動かなくなる

表の幅をうまく設定する方法を知っている方がいれば教えてください。

おまけ、2.4-2.7に関しても振り返ってみた

- 5.起動するまで壊れていることに気づかない(CI)
- 6.他メンバーの修正を上書きして消してしまう(上書き)
- 7.自信を持ってリファクタリングできない(リファクタ)
- 8.バグの修正時期がわからず、デグレートも追跡できない(デグレ追跡)
- 9.ブランチ・タグを活用できていない(ブランチ・タグ)
- 10.テスト環境や本番環境では動かない(管理システム)
- 11.リリースが複雑で手順書が必要となる(リリース作業)

内容	K●B●	コ○ボ
5.CI_	実機テストを手動で実行	リリース前に手動テスト実行
6.上書き	目視で回避(開発メンバーは2人)	masterをマージすると動かない機能があつたりした
7.リファクタ	出来るだけしない方針	出来るだけしない方針
8.デグレ追跡	NotesDBのレビュー文書	gitの履歴

内容	K●B●	コ○ボ
9.ブランチ・タグ	リリースの度にタグ付け_____	リリースの度にタグ付け_____
10.管理システム	静的リンクでバイナリ作成	mavenで管理
11.リリース作業 _____	コマンド一つで可能	AWS上にファイルアップロード

## 今後の課題

- 色々使ってみる やった方がいいとわかっていても、経験がないと仕事で実施できない
- 自分で環境構築して使えるようになる 個人の簡単なプログラムとかでも使ってみる
- 素早くテストコードを書けるようになる 書くと機能実装するだけより3~4倍時間がかかる