テストケースの 技術的負債との付き合い方

急成長SaaSの裏で、テストケース管理をどう改善してきたか？

リンクアンドモチベーションQAチーム 小島直毅

みなさん、
テストケースを
どのように管理して
いますか？

テストケース管理方法に
理想はありますか？

「最良のツール」は存在しない。最良の選択を。

Today's Message:

課題設定技術こそ技術者の本領。
そしてQAエンジニアリングの1番のおもしろさ！

Why, me?

激変の中で変わり続ける課題を体感してきたから

SaaS上場企業 ARRランキング 2020/11 - 企業データが使えるノート

順位	企業	対象	ARR (億円)	ARR成長率 (% / YoY)
14位	リンクアンドモチベーション	モチベーションクラウドシリーズ	22.8	10.7

有数のSaaSプロダクトに成長

開発チームが短期で激変

色々な品質課題を呼んだ「組織拡大」

結合観点の
漏れ

人が把握
できない

ナレッジの
分散

階層間で
連携ミス

組織拡大！

属人性増加

暗黙知の増加

組織拡大しながら、 高速・高品質に顧客価値を届けたい！

成功や失敗から得た学びを、技術的負債の観点からお話しします

苦労話や実際の資料を元に、
テストケース管理を変えていった
過程を共有します

そろそろ
自己紹介も
させてください

—

小島 直毅 Naoki Kojima

これまで

英語ディベート
青春を全て投げ捨てる

途上国開発の研究
紛争やテロ後の平和構築

エンジニア
テックで労働課題を解きたい

所属・活動

SigSQA

大学時代は札幌！

**MOTIVATION
CLOUD**
by Link and Motivation Group

年間収益

20億円

QAメンバー数

4 / 40人

提供サービスの概要

ToB 向け サービス

モチベーションクラウドシリーズ
組織の診断・改善を実現するクラウドサービス

診断

改善

ToC向け サービス

i-Company CLUB
「自分磨きの習慣」を
サポートするクラウドサービス

i-Company CLUB
by Link and Motivation Group

※再掲

組織拡大しながら、
高速・高品質に顧客価値を届けたい！

成功や失敗から得た学びを、技術的負債の観点からお話しします

そもそも、技術的負債とは？

そもそも、技術的負債とは？

「早さ」を求めて蓄積された課題が、
徐々に開発速度を遅くしていく現象

例: 不必要に複雑な設計やコード、アップデートされていないミドルウェア、
低カバレッジなテストコード、共有されていないナレッジ

テストの技術的負債とは？

「早さ」を求めて蓄積された課題が、
徐々に**テスト速度**を遅くしていく現象

例: 使い捨てられたテストケース、統一されていないテストケース管理、
共有されていないナレッジ

=> 本日は、テスト活動を遅くする現象を
「テストの技術的負債」として扱います

テストの技術的負債の具体的イメージ

使い捨てたテストケース

共有・更新できていない仕様

急成長する新規プロダクトの場合、
テストの技術的負債は
どう蓄積されるのか？

技術負債の移り変わり

技術的
負債

増加中

課題が表面化

返済期

最適化

立ち上げから、技術的負債が表面化するまで

立ち上げ

増加中

多チーム化

課題が表面化

ナレッジの活用

返済期

品質文化の拡大

最適化

技術的
負債

理想的な技術的負債のイメージ

新規プロダクトの技術的負債

こんなに上手くはいきませんでした

立ち上げ

負債

返済

安定

新規製品の技術的負債

初期はとにかく
スピード重視

新規製品の技術的負債

人数増加や
多チーム化

初期はとにかく
スピード重視

新規製品の技術的負債

人数増加や
多チーム化

じわじわと
雲行きが怪しく...

初期はとにかく
スピード重視

新規製品の技術的負債

技術的負債の蓄積を察知するのは困難です

もし察知できても、内部品質を上げるアクションを
ロードマップに中々盛り込めません

その結果...

じわじわと
課題が顕在化

多チーム化

初期はとにかく
スピード重視

30

20

10

0

変化の積み重ね

実際に発生した問題

PJT炎上・
リリース延期発生

最大 **3** 件/月

インシデントの件数

エンゲージメント
スコア大幅 $\downarrow$

みんなが自信を失っていく

どんな問題が起きていたか？

顕在課題

潜在課題

どんな問題が起きていたか？

テスト設計工数

結合観点のインシデント

顕在課題

潜在課題

どんな問題が起きていたか？

テスト設計工数 ↑

顕在課題

結合観点のインシデント ↑

未来の新メンバーを迎えられない

潜在課題

どんな問題が起きていたか？

では、何から手をつければいいのか？

仕様勉強会、テスト自動化、テスト観点集、など、
アイデアはたくさん出るが...

テスト設計工数 ↑

結合観点のインシデント ↑

未来の新メンバーを迎えられない

顕在課題

潜在課題

着眼点: テストの技術的負債

「早さ」を求めて蓄積された課題が、
徐々に**テスト速度を遅くしていく**現象

遅れれば遅れるほど、
納期に対するプレッシャーや品質後回し文化につながる

どんな問題が起きていたか？

テスト設計工数

顕在課題

結合観点のインシデント

未来の新メンバーを迎えられない

潜在課題

急いだ結果、フォームが崩れる

属人化している仕様
使い捨てられたテストケース

テスト設計工数 ↑ 結合観点のインシデント ↑

顕在課題

プロジェクト個別最適な文化

未来の新メンバーを迎えられない

潜在課題

まとめ

技術的負債で
テスト速度 ↓

納期に焦るばかり、
テストケース管理が疎かに

抜け漏れから
インシデント発生

まとめ

プロジェクト個別最適で、
前後の時間軸への配慮がない

全体のことを考え、
前後の時間軸に配慮する

技術的負債で
テスト速度↓

テストケース管理が疎かに

抜け漏れから
インシデント発生

「テストケースを育てる」文化をつくる！

仕様を蓄積するため、テストケース管理改善からはじめました

Then, how?

よし！早速、テストケースを改造だ！

No.	仕様書ID	項目	内容	テスト項目	優先度	備考
1	仕様書ID	項目	内容	テスト項目	優先度	備考
2	仕様書ID	項目	内容	テスト項目	優先度	備考
3	仕様書ID	項目	内容	テスト項目	優先度	備考
4	仕様書ID	項目	内容	テスト項目	優先度	備考
5	仕様書ID	項目	内容	テスト項目	優先度	備考
6	仕様書ID	項目	内容	テスト項目	優先度	備考
7	仕様書ID	項目	内容	テスト項目	優先度	備考
8	仕様書ID	項目	内容	テスト項目	優先度	備考

よし！早速、テストケースを改造だ！

Test Case ID	Test Case Description	Test Case Data	Test Case Result
TC001	Test Case 1	Test Case Data 1	Test Case Result 1
TC002	Test Case 2	Test Case Data 2	Test Case Result 2
TC003	Test Case 3	Test Case Data 3	Test Case Result 3
TC004	Test Case 4	Test Case Data 4	Test Case Result 4
TC005	Test Case 5	Test Case Data 5	Test Case Result 5
TC006	Test Case 6	Test Case Data 6	Test Case Result 6
TC007	Test Case 7	Test Case Data 7	Test Case Result 7
TC008	Test Case 8	Test Case Data 8	Test Case Result 8
TC009	Test Case 9	Test Case Data 9	Test Case Result 9
TC010	Test Case 10	Test Case Data 10	Test Case Result 10

Today's Messages:

課題設定技術こそ技術者の本領。
そしてQAエンジニアリングの1番のおもしろさ！

まずは課題から

何度も改善・テスト

SaaSなのでケースを長く使いたい！
(とはいえすぐには自動化できない)

データが大事なHRtech

エンジニアが詳細まで
テストできるようにしたい！

将来も見据えた分析

「テストが育ち、誰にも使われ、かつメンテされ続ける」
ことが必要と判断。

何度も改善・テスト
ケースを長く使いたい！

データが大事なHRtech
エンジニアが詳細まで
テストできるようにしたい！

組織拡大のための文化戦略

スケーラブルな
組織構造へ

「テストケースを育てる」文化の浸透

どうやるか？

仕組み化

文化組織の自走

スケーラブルな
組織構造へ

「テストケースを育てる」文化の浸透

どうやるか？

属人性

仕組みへの参入障壁

仕組み化

文化組織の自走

スケーラブルな
組織構造へ

「テストケースを育てる」文化の浸透

どうやるか？

属人性 ↓

ドメイン知識依存の脱却

仕組み化

みんながテストできる
仕組みをつくる

スケーラブルな
組織構造へ

「テストケースを育てる」文化の浸透

属人性 ↓

ドメイン知識依存の脱却

仕組み化

みんながテストできる
仕組みをつくる

スケーラブルな
組織構造へ

「テストケースを育てる」文化の浸透

簡単にはいかないテストケースの一元化

バラバラでわからない

誰でもわかる！

元ファイルの整理と成果物の定義から

- テストレベル順

- 機能別

- 時系列順

スピード重視で整理！

プロダクトの画面と
同じ階層構造で整理！

テストケースのフォルダ構成設計

Before

After

うまくいかないテストケースの統合

項目	内容	担当者	ステータス	優先度	備考
テストケース1	機能Aの動作確認	担当者A	実行済み	高	エラー発生
テストケース2	機能Bの動作確認	担当者B	実行済み	中	エラー発生
テストケース3	機能Cの動作確認	担当者C	実行済み	低	エラー発生
テストケース4	機能Dの動作確認	担当者D	実行済み	高	エラー発生

項目	内容	担当者	ステータス	優先度	備考
テストケース5	機能Eの動作確認	担当者E	実行済み	高	エラー発生
テストケース6	機能Fの動作確認	担当者F	実行済み	中	エラー発生
テストケース7	機能Gの動作確認	担当者G	実行済み	低	エラー発生

Before

担当者やPJT最適な分散構造

項目	内容	担当者	ステータス	優先度	備考
1	機能Aの動作確認	担当者A	実行済み	高	エラー発生
2	機能Bの動作確認	担当者B	実行済み	中	エラー発生
3	機能Cの動作確認	担当者C	実行済み	低	エラー発生
4	機能Dの動作確認	担当者D	実行済み	高	エラー発生
5	機能Eの動作確認	担当者E	実行済み	高	エラー発生
6	機能Fの動作確認	担当者F	実行済み	中	エラー発生
7	機能Gの動作確認	担当者G	実行済み	低	エラー発生

After

新人にもわかる一本構造

テストケース管理の枠が完成！

プロダクト ➤ アーキテクチャ ➤ 機能 ➤ 画面 ➤ テストケース

属人性 ↓

ドメイン知識依存の脱却

仕組み化

みんながテストできる
仕組みをつくる

スケーラブルな
組織構造へ

「テストケースを育てる」文化の浸透

テスト工数不足 → 小さなテストはエンジニアが自走！

新規開発・大規模修正

実装

QAがテスト

リリース

テスト工数不足 → 小さなテストはエンジニアが自走！

新規開発・大規模修正

実装

QAがテスト

リリース

小中規模改修

実装

エンジニアがテスト

リリース

エンジニアテスト普及までの苦勞

🔗 エンジニアテスト設計書URL

※設計書は以下のフォルダに格納をお願いします

PRテンプレート

slack botのGitHub連携

エンジニアをプロセス策定・
全体共有・事例作りに巻き込む

若手でもわかる手順を整える

意識せずとも使える
仕組みに落とす

属人性 ↓

ドメイン知識依存の脱却

仕組み化

みんながテストできる
仕組みをつくる

スケーラブルな
組織構造へ

「テストケースを育てる」文化の浸透

仕組みを文化にする

見えてきた将来像

スケーラブルな組織 = 技術的負債を継続的に解消できる

スケーラブルな組織 = 技術的負債を継続的に解消できる

組織文化は1日にして成らず
されど、組織文化なしに安定拡大ならず

積み重ねた結果

最大 **8** 割減

既存機能のテスト設計工数

3件/月 => **0.3**件/月

インシデント件数

0% => **30%**

エンジニアのテスト設計割合

品質文化が立ち上がり、さらなる進化へ

立ち上げ

増加中

多チーム化

課題が表面化

ナレッジの活用

返済期

品質文化の拡大

最適化

技術的
負債

これからの課題: テストケースの保守性向上へ

「テストケースはプロダクトの一部」
という認知を広めたい

バージョン管理や更新プロセス
を強化したい

「テストケースを育てる」文化をつくる

最良の選択を。

最後に:

課題設定技術こそ技術者の本領。
そしてQAエンジニアリングの1番のおもしろさ!

現状を常に正しく把握するためにも、
見えにくいテストの技術的負債を意識していきましょう。

wantedlyでQA、エンジニアなどなど募集中です！

募集 ストーリー

株式会社リンクアンドモチベーション

フォロー中

<http://www.lmi.ne.jp> 東京都

ホーム 私たちについて メンバー ストーリー 募集

この後は...

**会場4F 研修室402、
discordの「情報交換会：事例：小島」にあります！**

スライドに乗せきれなかった詳細な資料や、
エンジニアを巻き込み文化をつくるポイントなど
お話しできたら嬉しいです