Ameba

受け入れテスト「７名体制」で年間１００タイトルをこなすためには！

株式会社サイバーエージェント
アメーバ事業本部
川上 琢也 (カワカミタクヤ)

自己紹介

川上 琢也 (カワカミタクヤ)

株式会社サイバーエージェント

アメーバ事業本部 経営本部部門

メディアサポート室 **SMAqグループ**

品質管理
責任者

@takubon

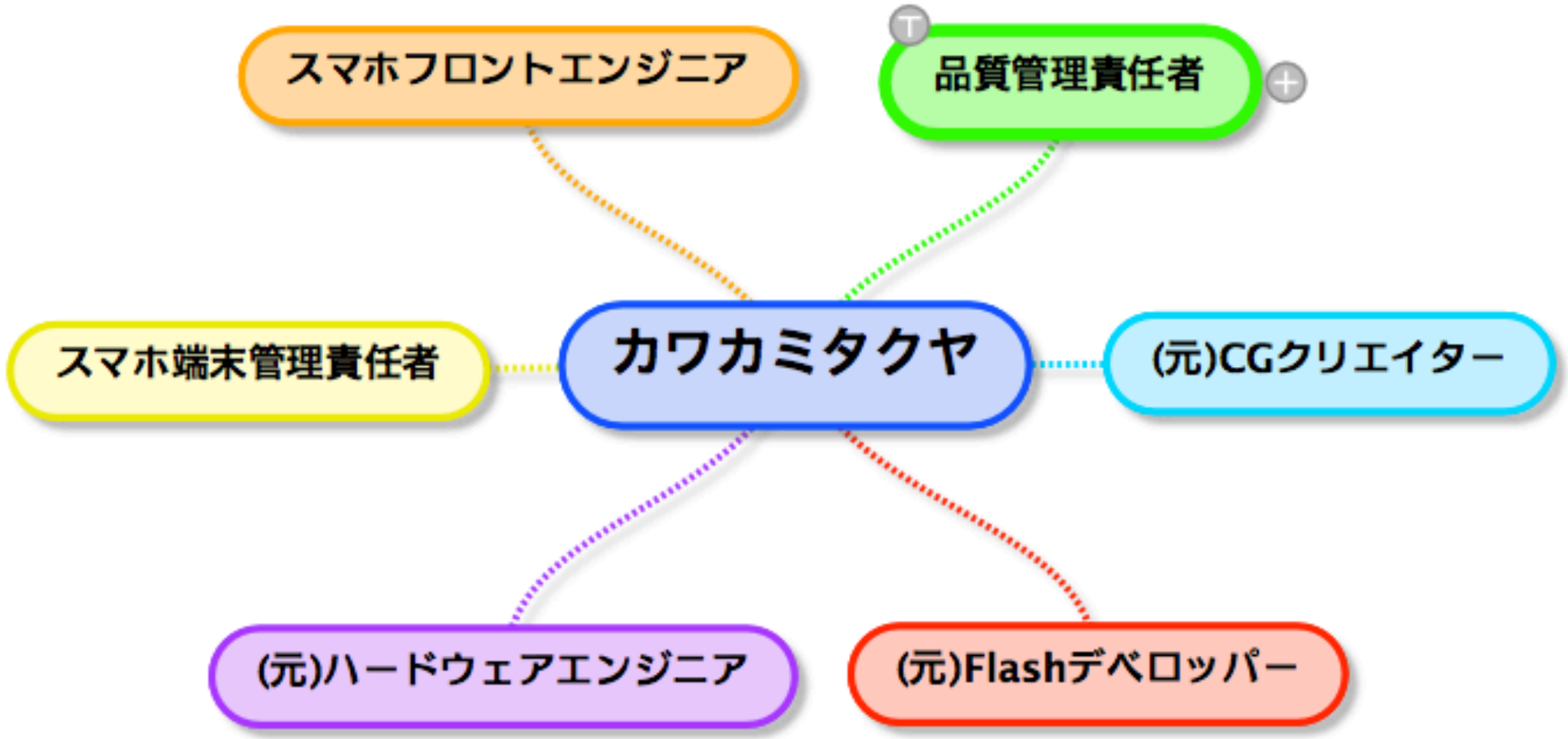
publicbath

kawakami_takuya@cyberagent.co.jp

自己紹介

コインプラザ

SMAq
検証司令塔



2年半前に中途入社、1年間ほどスマホアプリを開発し
品質管理グループ(SMAq)を立ち上げ、現在に至ります。

アジェンダ

・Amebaスマホサービスについて
・テストの体制
・課題と解決
・成功事例
・これからの取り組み
・質疑応答

技術よりの話は
一切しないので
ご了承下さい。

まず初めに
Amebaのスマホ事業を
ご紹介します

Amebaスマホサービス

AmebaスマホTVCM

AmebaスマホTVCM

AmebaスマホTVCM

AmebaスマホTVCM

Amebaスマホサービス

Amebaでは

- ゲーム
- コミュニティ
- ホーム

など
様々なスマホアプリを
内製で開発しています

● 年間１００タイトル

およそ

その数なんと「年間１００タイトル」

● 年間１００タイトル

今回のテーマ

たった７名で全アプリのテストを受け入れることが可能なのか？

本題に入る前に

みなさんにアンケート

バリエーションテストは誰が実施していますか？

1．開発チームの**メンバー自ら**実施
もしくはチーム内の**テスト担当者**が実施

2．品質管理部門(**QA部門**)が実施

3．**デバッグ会社**に委託して実施

4．バリエーションテストを**やっていない**

Amebaの
バリエーションテスト

バリエーションテスト

Amebaのバリエーションテスト

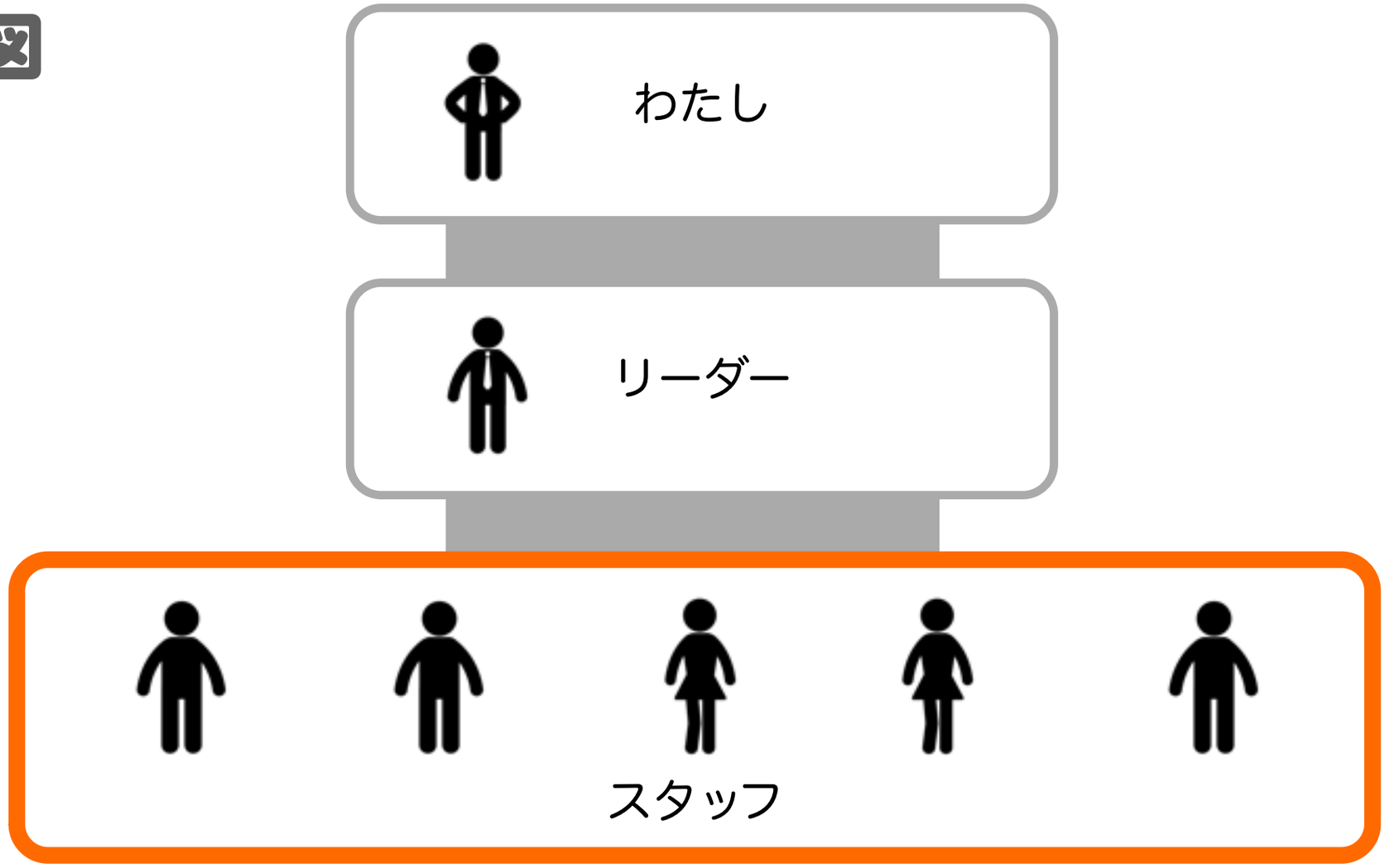
- **端末：8～12端末** にて実施
- **実施：外部デバッグ会社**に依頼して
  テスト結果を元に品質判定
- **日数：約7営業日** (実施5日、修正確認2日)

これを踏まえた上で…

テスト体制の話

テスト体制

体制図



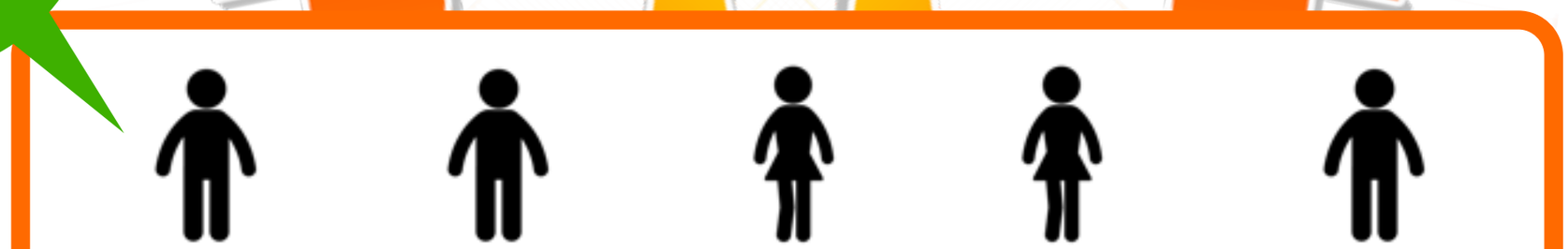
テスト対応スタッフは実質５名

テスト体制

体制図

各プロジェクト

テスト実施のコンサルタント



スタッフ

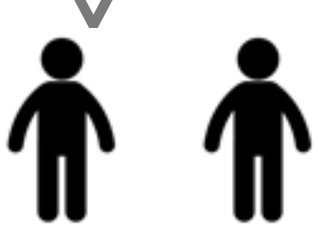
A社	B社	C社	D社
デバッグ会社	デバッグ会社	デバッグ会社	デバッグ会社

テスト体制の変化

立ち上げ

立ち上げ

落ち着いています



２名体制

Amebaアプリのテストを対応

● 3ヶ月経過

3ヶ月経過

まだ落ち着いています

4名に増員
月1本ペース

スマホアプリ全般対応

● 6ヶ月経過

6ヶ月経過

少し負荷が高くなり始めた

6名に増員
月2～3本ペース

Amebaがスマホに注力
開発体制80ラインへ
リリースはまだ先

１２ヶ月経過

１２ヶ月経過

高負荷で全体がバタバタ

７名に増員
月１０本ペース

TVCMに合わせ
テストラッシュ

そして現在

現　在

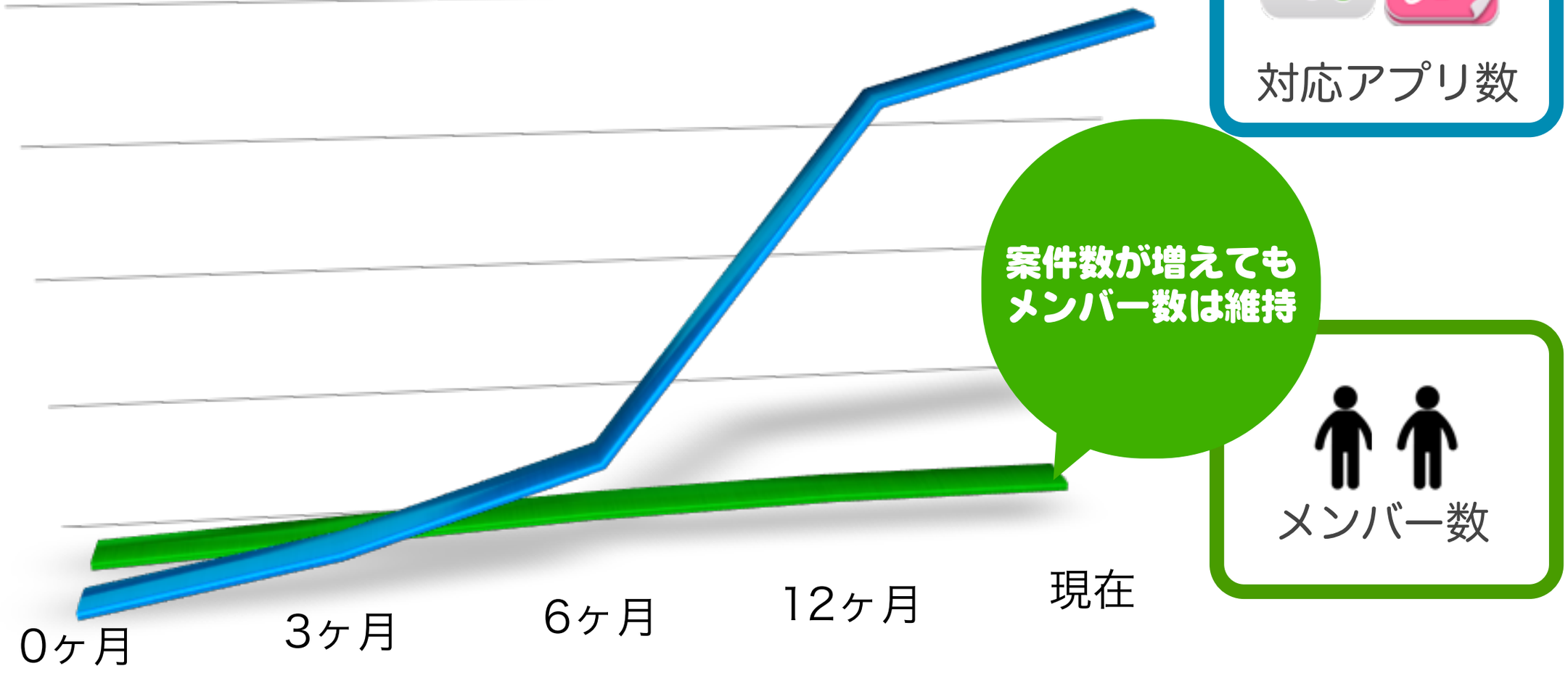
ようやく安定

7名体制を維持
月12本ペース

80ラインのアプリ
がひっきりなしに
依頼がくる状態

メンバーとアプリの推移

時系列にグラフ化してみました



メンバー数と対応アプリの推移グラフ

課題だらけの状況

課題だらけの状況

案件数が急増！

課題だらけの状況

テストメンバー数が

ほぼ現状維持！

１つのテストにかける

時間を短縮するしかない！

手を抜いて品質を

落とすわけにはいかない！

課題点

課題点をまとめてみると…

- 最小メンバーで最大限の案件をこなす
- テスト実施期間の短縮
- テストのシンプル化
- アプリの品質は下げない

どうやって解決したのか？

徹底的に無駄を省き

片っ端から統一化した

課題解決
の
実例を紹介します

課題１（報告手段の不明確性）

報告手段が決まっていない 課題

・この報告はどのように報告すればよいか決まっていない

・テスト初日に問い合わせラッシュでテストが進まない

報告手段のルール化 解決

・報告内容と緊急度に応じて使い分ける指標を設けた

・メール、redmine、Skype、電話 (※右ほど緊急度高)

課題２（リアルタイム性の欠如）

リアルタイム性の欠如　課題

・Excel形式のバグ報告が纏まってくる(日次レポート)

・テスト期間にリアルタイムでバグ修正ができない

redmineの利用　解決

・リアルタイムにバグ報告ができる

・ウォッチャーにプロジェクトのMLを入れ、バグ報告を拡散

・テスト期間に裏側での修正対応が可能に

課題３（バグ報告のばらつき）

バグ報告のばらつき　課題

・チケット記載方法がデバッグ会社によって異なる

・説明文や再現手順が不明確で理解できない、再現できない

・キャプチャー画像、デバッグログがない

バグ報告の統一化　解決

後ほど紹介

・チケット記載フォーマットを各社統一

・ログ、キャプチャー(画像、**ムービー**)を添付

・新規チケット作成時のひな形作成ツールを導入(プラグイン)

課題３（バグ報告ひな形）

参考：バグ報告ひな形

■バグ内容

■手順

■設定条件

■端末

■アプリver

■発生時間

■頻度

■ID

■ログ・キャプチャ

ステータス: 終了
優先度: 通常
担当者:
カテゴリ: -
対象バージョン:
次回バージョンアップ対応: いいえ
端末固有: いいえ

説明

■バグ内容
プロフィール編集画面にて、禁止文字を入力し「決定」を押下後、
入力エラーのポップアップが表示されますが、「はい」をタップしてもポップアップが消えません。

■手順
1.プロフィール編集画面へ遷移する
2.禁止文字を入力する
3.「決定」を押下する

■設定条件
禁止文字を入力する

■端末

■アプリver

■発生時間
2013-02-14 17:11:42

■頻度
5 / 5

■ID

バグチケット

課題４（バグ深刻度のばらつき）

バグ深刻度のばらつき　課題

・致命的なバグなのか？優先度の低いバグなのか？ 深刻度が分からない

・プロジェクト側はどのバグから修正すれば良いか分からない
必須対応か？次回バージョンアップで対応で良いのか？

深刻度の判断指標を設けた　解決

・バグ深刻度の判断基準表を作成し、判断を容易にできるようにした
→ 判断する担当者の差異を吸収

・指標には具体例を記載し、当てはまらない場合は判断観点にて対応

課題４（バグ深刻度基準表）

参考：バグ深刻度基準表

■カテゴリー別：バグ深刻度基準（概要）

深刻度	説明	カテゴリ	対応
高め	ユーザの操作や進行、理解を妨げる	・クラッシュ ・フリーズ ・データ破損 ・ログイン ・ユーザ情報 ・課金 ・機能(粗幹)	修正が必須 ※修正されない場合はリリース不可
通常	ユーザの操作や進行、理解に著しく影響する	・UI関連(重度) ・機能(アプリ) ・機種依存(重度) ・サウンドの欠落 ・設定の反映 ・安定性 ・パフォーマンス ・エラー処理	すぐに修正が必要 ※新規なら2ヶ月以内、既存なら必須
低め	ユーザの操作や進行、理解に影響を与えない	・UI関連(軽度) ・誤字 ・レアケース ・機種依存(軽度)	任意で修正

概要

■現象・事例別：バグ深刻度基準（詳細）

バグ深刻度	カテゴリ	タイトル	現象・事例
高め	ユーザ情報	ユーザ情報取得不可	・ユーザ情報が取得できない
高め	ユーザ情報	ユーザ情報不整合	・他人の情報が反映される ・取得データが反映されない ・取得データが前画面と混在している(混ざる)
高め	ユーザ情報	ユーザ情報設定不能	・ユーザ情報の設定ができない ・ユーザ情報の変更が反映されない （プロフィール、サムネイルなど）
高め	ログイン	ログイン不可	・ログインに失敗する ・ログイン完了後に画面が遷移しない
高め	ログイン	ログアウト不可	・ログアウトに失敗する ・ログアウトが完了してもログイン情報が引き継がれる
高め	ログイン	ユーザ切替不可	・ログアウト→ログインのフローで動作に異常がある
高め	ログイン	ログイン情報異常	・アプリ起動時に毎回ログインを要求される ・関連アプリ起動時にログインを要求される ・関連アプリ間でログイン情報が引き継げない ・関連アプリ間でログイン情報が異なる
高め	ログイン	会員登録異常	・会員登録画面が表示されない ・メールが配信しない ・仮メールが送れない ・登録フォームが表示されない ・登録後にアプリに戻れない ・登録が完了しますと表示しない
高め	ログイン	非会員データ引き継ぎ異常	・非会員→会員時にデータがクリアされる ・非会員→会員の切り替えでデータの複製ができる
高め	クラッシュ	起動不可	・アプリが起動しない ・起動直後に落ちる
高め	クラッシュ	落ちる	・利用中に落ちる NullPointerException(Android)
高め	クラッシュ	デバイス依存	・操作を受け付けない ・無限ロードする ・異常な低速処理
高め	クラッシュ	描画不可	・真っ黒画面 ・真っ白画面
高め	フリーズ	フリーズ	・利用中にフリーズする
高め	フリーズ	サーバーエラー	・Internal Server Error ・サーバーエラー発生後に動作不能となる
高め	データ破損	データ保存異常	・データが保存できない ・保存したデータに不備がある
高め	データ破損	データ喪失	・保存したデータが消える ・ユーザを切り替えると前ユーザのデータが上書かれる
高め	課金	購入不可	・アプリの購入(加算)ができない ・デココインの購入・交換(加算)ができない ・加算後の残高がおかしい ・各種課金決裁のフローが成立しない （Apple, Google, キャリア, PostPayなど）
高め	課金	消費不可	・アプリ内消費（減算）ができない ・デココインの消費（減算）ができない ・消費後の残高がおかしい ・アプリ内コインでアイテムが購入できない
高め	課金	アイテム利用不可	・購入したアイテムが利用できない
高め	課金	アイテム反映異常	・購入アイテムが反映されない
高め	機能(複数)	データストレージ異常	・画像フォルダ内の画像一覧が表示されない ・画像フォルダ内の画像を取得・編集・アップロードできない ・指定フォルダ内の音楽一覧が表示されない ・指定フォルダ内の音楽を再生できない
高め	機能(複数)	カメラ機能異常	・カメラが起動しない ・撮影した画像が保存できない
高め	機能(複数)	センサー機能異常	・傾きセンサーが反応しない ・傾けた方向に表示が切り替わらない
高め	機能(複数)	キーボード異常	・キーボードが表示されない ・キーボード入力を受け付けない
通常	UI関連(重度)	表示崩れ	・表示が大きく崩れる ・表示位置がずれ過ぎていて見れない ・表示サイズが「大き過ぎる」もしくは「小さ過ぎる」 ・横画面／縦画面で表示内容が大きく異なる
通常	UI関連(重度)	ボタン表示不備	・ボタンがタップできない ・ボタンをタップしても反応しない ・ボタンのサイズが極めて小さい ・ボタンの表示位置が大きくずれている ・ボタンの位置とタップエリアの位置が大きくずれている ・下層で隠れているボタンをタップできる
通常	UI関連(重度)	テキスト不備	・文字化け ・テキストの視認性が極めて悪い ・フォントサイズが「大き過ぎる」し「小さ過ぎる」 ・フォントカラーが適切されず背景色と同化する
通常	機能(アプリ)	写真機能異常	・撮影した写真の品質が極めて悪い ・写真が投稿できない ・投稿した写真の画質が極めて悪い（画質の劣化） ・投稿した写真の縦横が逆になる ・投稿した写真の編集ができない
通常	機能(アプリ)	メッセージ投稿機能異常	・メッセージが投稿できない ・投稿した内容が反映されない ・投稿したメッセージが一部欠落している
通常	機能(アプリ)	タイムライン異常	・タイムラインが表示されない ・タイムラインの表示順が時系列になっていない ・投稿した内容（他人・自分）がタイムラインに正しく表示されない
通常	機能(アプリ)	招待機能異常	・招待できない ・フォローできない ・招待の承認をしてもつながらない ならいね！できない ならいね！自動送信できない
通常	機能(アプリ)	プッシュ通知機能異常	・プッシュ通知が飛ばない ・プッシュ通知を受け取れない ・バッジが表示されない
通常	機能(アプリ)	画像フィルター機能異常	・指定したフィルターと違った効果がかかる ・画像が荒い ・明度が低過ぎる(暗過ぎる)、高過ぎる(白く飛ぶ) ・コントラストが高過ぎる、低過ぎる
			・操作できなくなる

具体例

課題５（項目書のばらつき）

項目書のばらつき　課題

・項目書を作成するメンバーによって記載ルールがばらばら

・項目書の内容を把握するのに時間を要する→生産性低下

・記載内容と仕様が合っていない場合が多い

項目書のひな形を作った　解決

・過去のテストから機能ごとに切り分け、汎用化的な雛形を作成
→ 項目書作成と添削時間の大幅短縮

・記入する担当者による差異を吸収できる
→ 項目書の品質向上、表記ゆれ防止

課題6（文言のばらつき）

文言のばらつき　課題

・同類アプリの同等機能のテスト項目で全く違う記載方法

・表記ゆれや誤字脱字が多い

「更新」≒「リロード」など文言が一致しない

共通言語化　解決

・機能とテスト観点が共通言語化することでテストがよりスムーズに

→ 質問数が激減、生産性向上

・テスターは深く読み取らなくてもテスト観点が理解できている

→ テスター負荷軽減、差し戻しバグチケット激減

課題7（チケット作成コスト肥大）

チケット作成コストの肥大　課題

・テスター側：バグチケットの説明文を記載するコストが大
　→ テスト生産性の低下

・プロジェクト側：内容を理解するコストが大

ムービーキャプチャの利用　解決

・バグ内容の説明文を短縮できる
　→ チケット作成コスト削減、生産性向上、質問数減

・動画だと一目瞭然
　→ 特にゲーム系のアニメーション、表示崩れで有効的

課題8（類似バグの多発）

類似バグの多発　課題

・以前に見た同種バグが他プロジェクトでも発生

・解決方法、発生原因の究明に時間を要する

ナレッジ共有　解決

・バグのカテゴリー分けし、展開した

端末依存型、OS依存型、独自ブラウザ依存型、メモリー依存型、キャリア依存型、
WebView依存型、海外端末依存型、カメラセンサー依存型、ローカルアクセス依存型

→ 同類バグを未然に防ぎ、バグ検出率もアップ

課題9（生産性のばらつき）

生産性のばらつき　課題

・見積もり通りにテストが完了しない場合がある

・固有の端末(テスター)で遅れを生じる場合がある

生産性を2倍に！？　解決

・通常はテスターと端末は1対1だが、2台にすれば単純に2倍！
→ 他の端末と比較できる為、バグ発見率がアップ

・進捗を日次ではなく、中間報告を入れる(テスト管理ツール導入)
→ 固有端末の遅れを早期発見、早期解決ができる

このように日々

ダカイゼン※

を続けています！

※ダカイゼンとは、「打開」と「改善」を
組み合わせたサイバーエージェントの造語です

取り組みの結果

成功例1（プロジェクトとの親和性）

プロジェクトとの親和性が向上しました

- QAを砦(トリデ)部隊にしない
  - 通常QAというと砦のイメージがあり、身構えられて対話ができない
  - コンサルタント(良き相談者)となり、品質向上を全面サポート
  - 各プロジェクトにあったテストプランを設計し、柔軟に対応する
- 親和性を高めた副産物
  - 開発や問題におけるノウハウ情報をいただける
  - 気軽にスケジュール遅れなどの相談してくれる

成功例2（振り返りミーティング）

ニーズにマッチ

- プロジェクトとの振り返りMTG
  - テスト実施後は必ずプロジェクト側から感想をヒアリングする
  - 要望が多い点は、即時カイゼン、即時反映
    - → 例：ムービーキャプチャ
- デバッグ会社との振り返りMTG
  - テスト実施における問題点を洗い出す
  - こちら側のやり方を全て押しつけない
  - よりスムーズに、よりシンプルに
    - → 例：redmineのチケット統一化

成功例３（ユーザの声）

感謝の声をいただけるようになった

▸スタッフのやりがい

このままリリースしていたらヤバかったです！

こんなことまでしてくれるんですか？

本当に助かりました！
心強いです。

今後の取り組み

（スコープの拡大）

スコープの拡大

今後の取り組みは…

- **設計、開発フェーズに入り込む**
  → 過去のバグ(ナレッジ)から、開発側にリスクを指摘

- **アジャイル開発におけるテスト**
  → スプリント毎テストだと工数肥大、スケジュール立て困難

- **リリース後の品質保証**
  → お問い合わせ内容を調査、受け入れテストにフィードバック

設計 → 開発 → 検証 → 運用
全フェーズでの品質向上を目指す！

結果はまたの機会に！

ご清聴
ありがとうございました

質疑応答タイム

川上 琢也 (カワカミタクヤ)

株式会社サイバーエージェント

アメーバ事業本部 経営本部部門

メディアサポート室 **SMAqグループ**

@takubon

publicbath

kawakami_takuya@cyberagent.co.jp

本日聞けなかった内容は
こちらまでご連絡下さい