割り勘支援アプリ テスト設計

チーム名：勇往米進

メンバー：佐護/原/小林/桑原/NABE

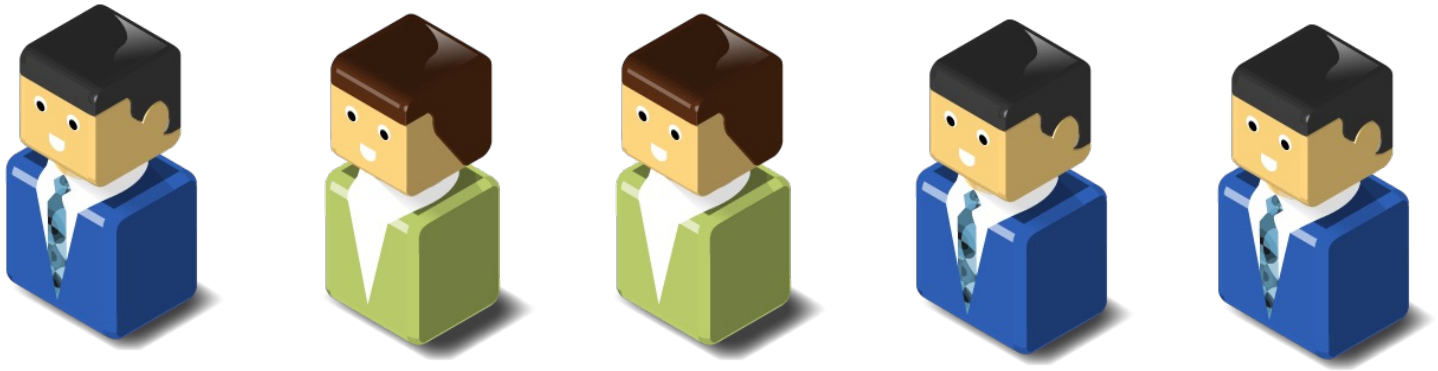
目次

1.チーム紹介

チーム構成　入社2～4年目でテスト設計に対し全員ほぼ初心者

チーム名　勇往米進

由来　勇往邁進（恐れることなく、目的に向かって前進しよう）
プラスで新潟で有名な米



佐護　原　小林　桑原　NABE

2.テスト概要

対象商品

割り勘を支援するためのスマホアプリ**Warikan**

テストの目的

1.アプリケーションが用途を満たしていることを確認
2.テストベースとソフトウェアのふるまいの合致性を検証
3.テストベースへ改善のフィードバックを行うこと

テスト設計方針

1.テストは全て手動でリリース版のアプリケーションを操作する形で行う。
2.テストは全てスクリプトテストとして作成する。
3.テスト実行は、テスト実行者2人で2日(1日6時間相当)以下で実施する。

参考：ASTER U30 テスト設計コンテスト テストプロジェクト要求補足書2022より

3-1.コンセプト

効率的なテスト設計をしていきたい！

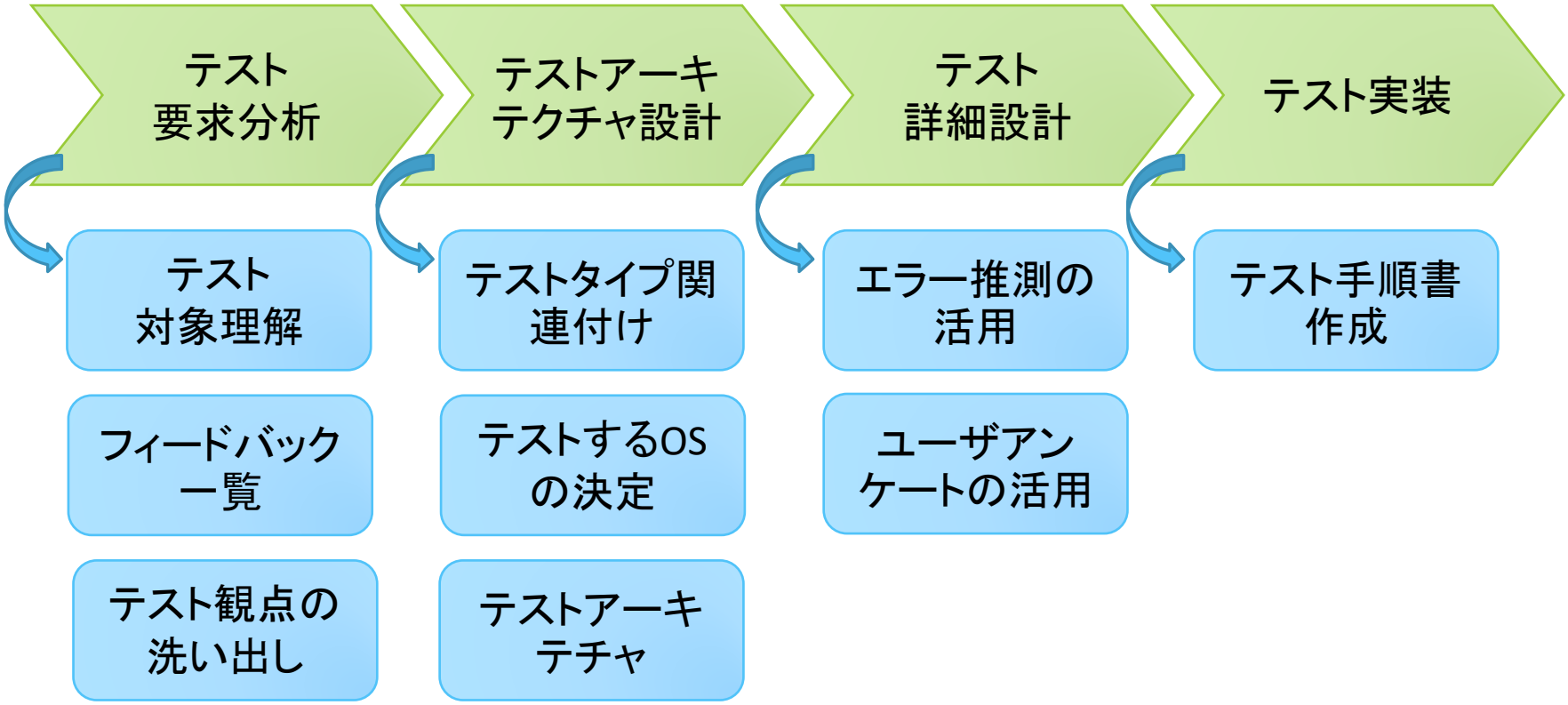
機能が適切に使用できることを保証したい！

割り勘で重要な計算機能は厚くテストをしていきたい！

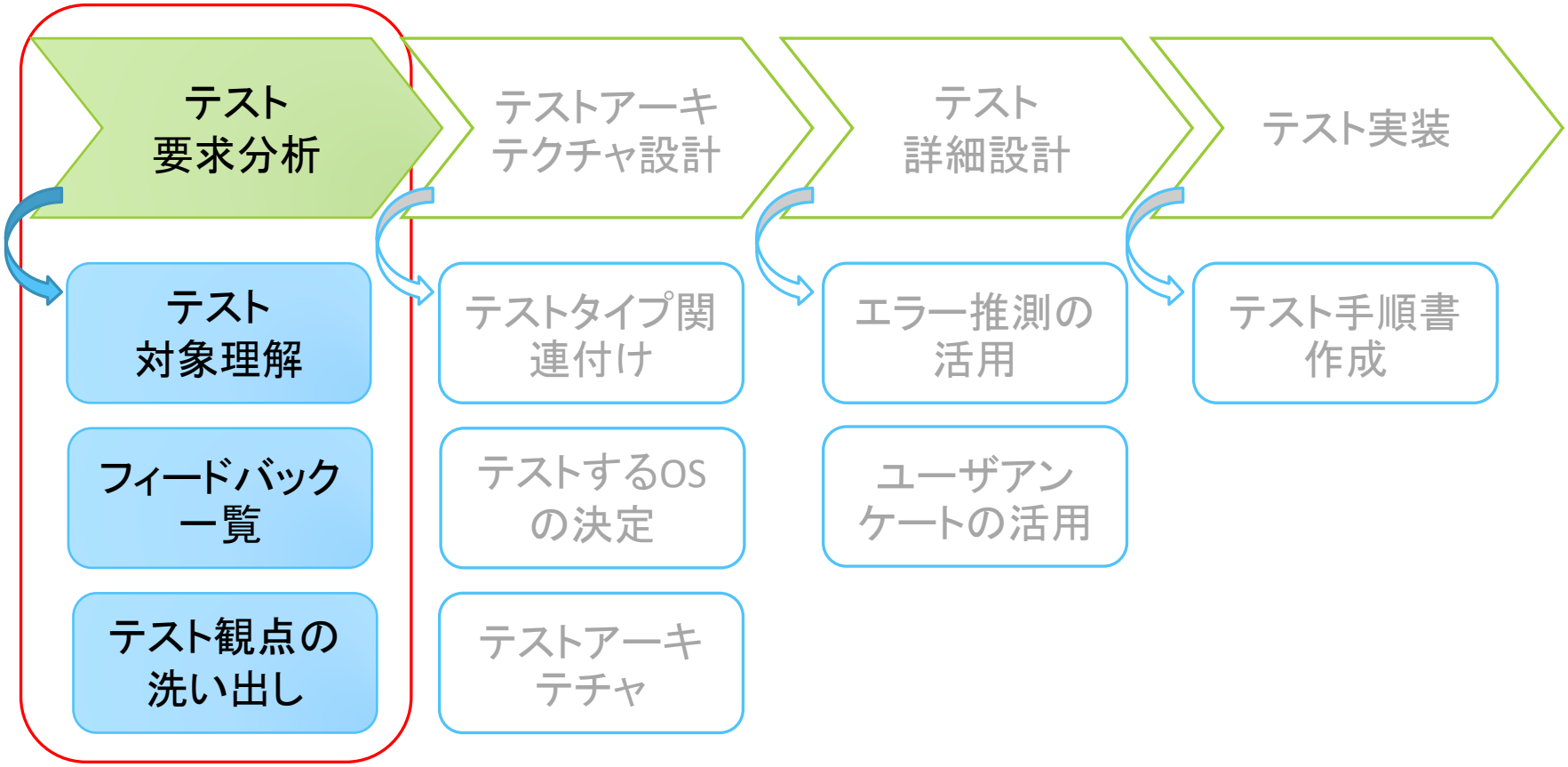
コンセプト

1.テスト実施工数に対して多くの欠陥を見つける
2.抜け漏れのないテストケースを作成する
3.割り勘の計算機能を重視したテスト設計を行う

3-2テスト設計のプロセス



3-3テスト要求分析


テスト
要求分析
テストアーキ
テクチャ設計
テスト
詳細設計
テスト実装
テスト
対象理解
フィードバック
一覧
テスト観点の
洗い出し
テストタイプ関
連付け
テストするOS
の決定
テストアーキ
テチャ
エラー推測の
活用
ユーザアン
ケートの活用
テスト手順書
作成

テスト対象理解

手順

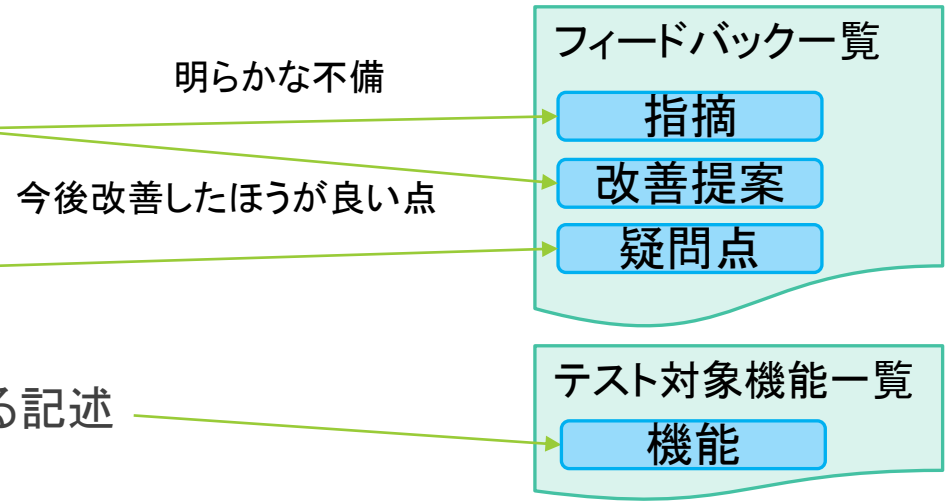
1.3色ボールペン法を用いてテストベースを読み込む（独自で定義）
2.赤と緑で線を引いた箇所をフィードバック一覧に書き込む
3.テスト依頼元からの回答を想定して記載（※本来は回答をもらう）
4.青で線を引いた箇所をテスト対象機能一覧として書き込む

ポイント

テストベースからテスト対象機能を抽出する
ことでソフトウェアのふるまいの合致性を検証

3色ボールペンの定義

- 赤 ： 仕様の不備
- 緑 ： 疑問点
- 青 ： 機能に関する記述



整理するため
3つに分類

フィードバック一覧

テストの目的

1.アプリケーションが用途を満たしていることを確認
2.テストベースとソフトウェアのふるまいの合致性を検証
3.テストベースへ改善のフィードバックを行うこと

フィードバック一覧
- 指摘
- 改善提案
- 疑問点

ポイント

テスト依頼元にもわかりやすいようにフィードバックの種類を分類して整理した。

No	資料	バージョン	仕様書の項目	分類	仕様書の記載	フィードバックの内容	テスト依頼元の回答	テスト設計での留意点
7	サンプルアプリケーション 割り勘支援アプリ Warikan仕様書	Ver. 1.0	2.4	改善提案	Warikan を終了すると、自動的にログアウト処理が行われます。	ログアウトするためのボタンを追加するのはどうか。	今後検討します	無し
8	サンプルアプリケーション 割り勘支援アプリ Warikan仕様書	Ver. 1.0	2.5	指摘	通信が不能だった場合、また通信に異常が発生した場合は、Warikan アプリケーションは「サービスサーバと正常に通信できません」とエラー通知	通信が不能または通信に異常が発生したと判断する方法の記載がない。	サービスサーバとの通信で10秒応答が無い場合には通信が不能または通信に異常が発生したと判断します。	サービスサーバとの通信で一定時間応答が無い場合を確認する。
9	サンプルアプリケーション 割り勘支援アプリ Warikan仕様書	Ver. 1.0	2.5	疑問点	その通信が不能だった場合、また通信に異常が発生した場合	サービスサーバとの通信が不能だった場合と、異常が発生した場合の違いは何か？	違いは以下の記載の通りです。 通信が不能：サーバーと通信できない状態。 通信が異常：サーバーとの通信中にエラーが発生した状態。	サーバーとの通信状態を機内モードに変更することでサーバーと通信できない状態を作り出しテストを実施する。

図：フィードバック一覧(一部抜粋)

テスト観点の洗い出し

洗い出し手順

1.抜け漏れのないテストケースを作成するために品質特性を活用
2.品質特性をベースにマインドマップで観点を出していく
3.マインドマップで出た観点を整理

テスト観点一覧を作成

ポイント

品質特性を用いることで抜け漏れを防止

・観点が対象外であるかどうかを検討する。

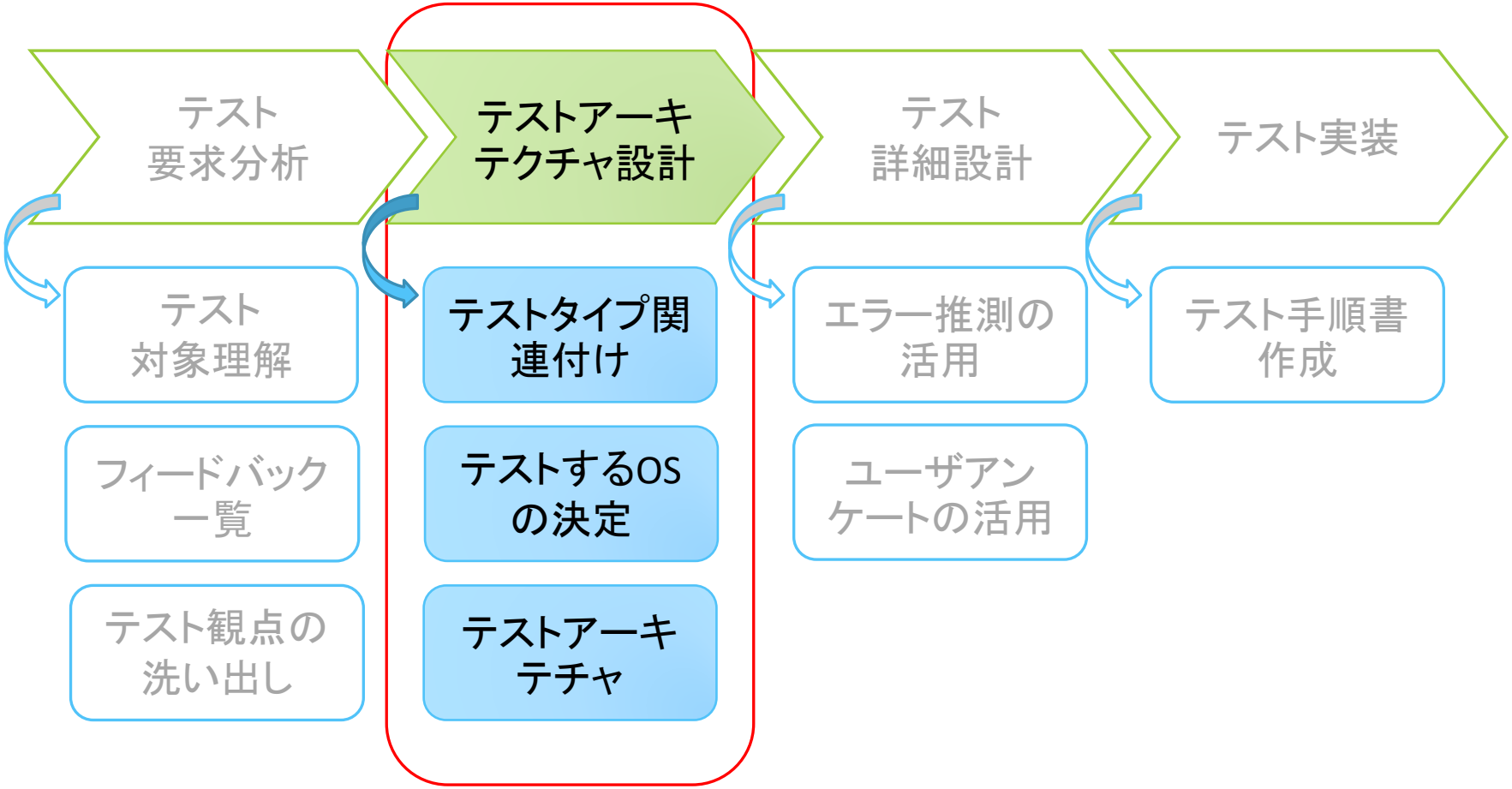
・市場への影響度を「高」「中」「低」の3段階に分けて優先度をつける。

テスト観点一覧

テスト観点ID	モデル	品質特性	品質副特性	定義	Warikan テスト観点	目的との合致	責務との合致	テスト対象	テスト対象外理由	機能/非機能	市場への影響度	備考
U-06	製品品質	使用性	運用操作性	製品又はシステムが，それらを運用操作しやすく，制御しやすくする属性をもっている度合い。	誤操作をしてもすぐに実施したい操作を行えるか確認する	○	○	対象	－	非機能テスト	高	－
U-07			ユーザエラー防止性	利用者が間違いを起こすことをシステムが防止する度合い。	誤った操作を防止する機能があるか確認する	○	○	対象	－		低	－
U-08			ユーザインタフェース快美性	ユーザインタフェースが，利用者にとって楽しく，満足のいく対話を可能にする度合い。	文字を認識しやすいか確認する	○	○	対象	－		低	－
U-09					ボタンを認識しやすいか確認する	○	○	対象	－		低	－
U-10					アプリは見やすいデザインか確認する	○	○	対象	－		低	－
U-11			アクセシビリティ	製品又はシステムが，明示された利用状況において，明示された目標を達成するために，幅広い範囲の心身特性及び能力の人々によって使用できる度合い。	高齢者が目的を達成できるか確認する	○	○	対象	－		中	－
U-12					視覚障害を有する人が目的を達成できるか確認する	○	○	対象	－		中	－
CR		信頼性		明示された時間帯で，明示された条件下に，システム，製品又は構成要素が明示された機能を実行する度合い。	－	－	－	－	－	－	－	－
CR-01			成熟性	通常の運用操作の下で，システム，製品又は構成要素が信頼性に対するニーズに合致している度合い。	正しい金額で送金処理が動作することを確認する	○	×	対象外	ジャスpayアプリは評価の責務の対象外となっているため対象外	－	－	－

図：テスト観点一覧(一部抜粋)

3-4テストアーキテクチャ設計



テストタイプ関連付け

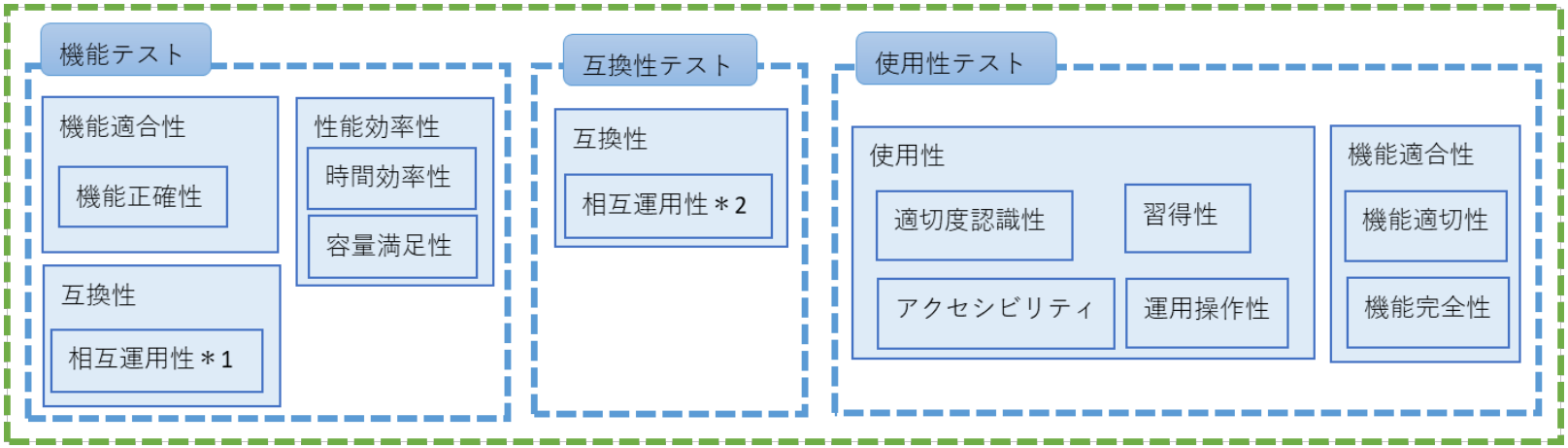
実施手順

1.テスト観点から関連のあるテストタイプを抽出
2.テスト観点をテストタイプ毎に分類

ID	テストタイプ
FT	機能テスト
UT	使用性テスト
CT	互換性テスト

ポイント

テストタイプ毎にまとめて整理することでテストの重複や抜け漏れを防げる



＊1　機能の動作を確認する観点

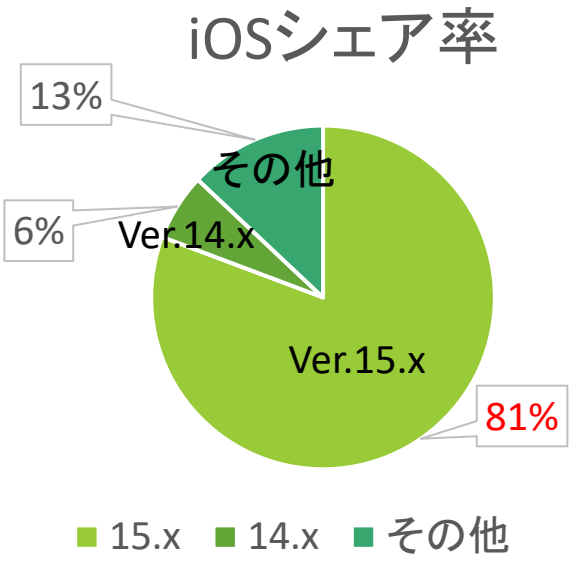
＊2　互換性の確認をする観点

図:テストタイプ関連付け(一部抜粋)

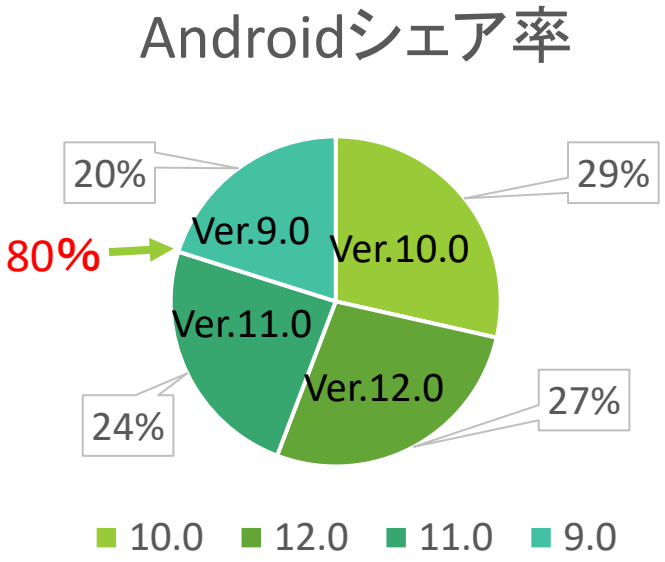
テストするOSの決定

手順

1.サポートされているOSでシェア率80%を保証する方針を決定
2.iOS/Androidそれぞれでシェア率を調べる
3.テストするOSを決定



テストするOS
15.x



テストするOS
10.0/11.0/12.0

図:OS確認方法 (一部抜粋)

テストアーキテクチャ

実施手順

1.リスクを考慮してテストの実施順を決定した。
2. テスト実行者が2人であることを考慮しテストするOSを振り分けて
テストアーキテクチャを作成

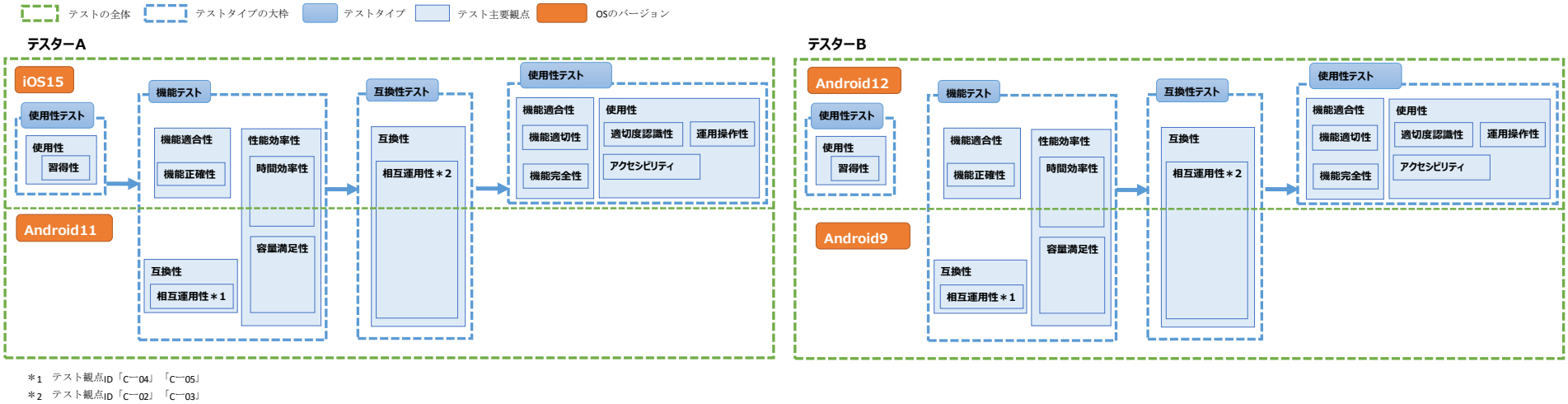
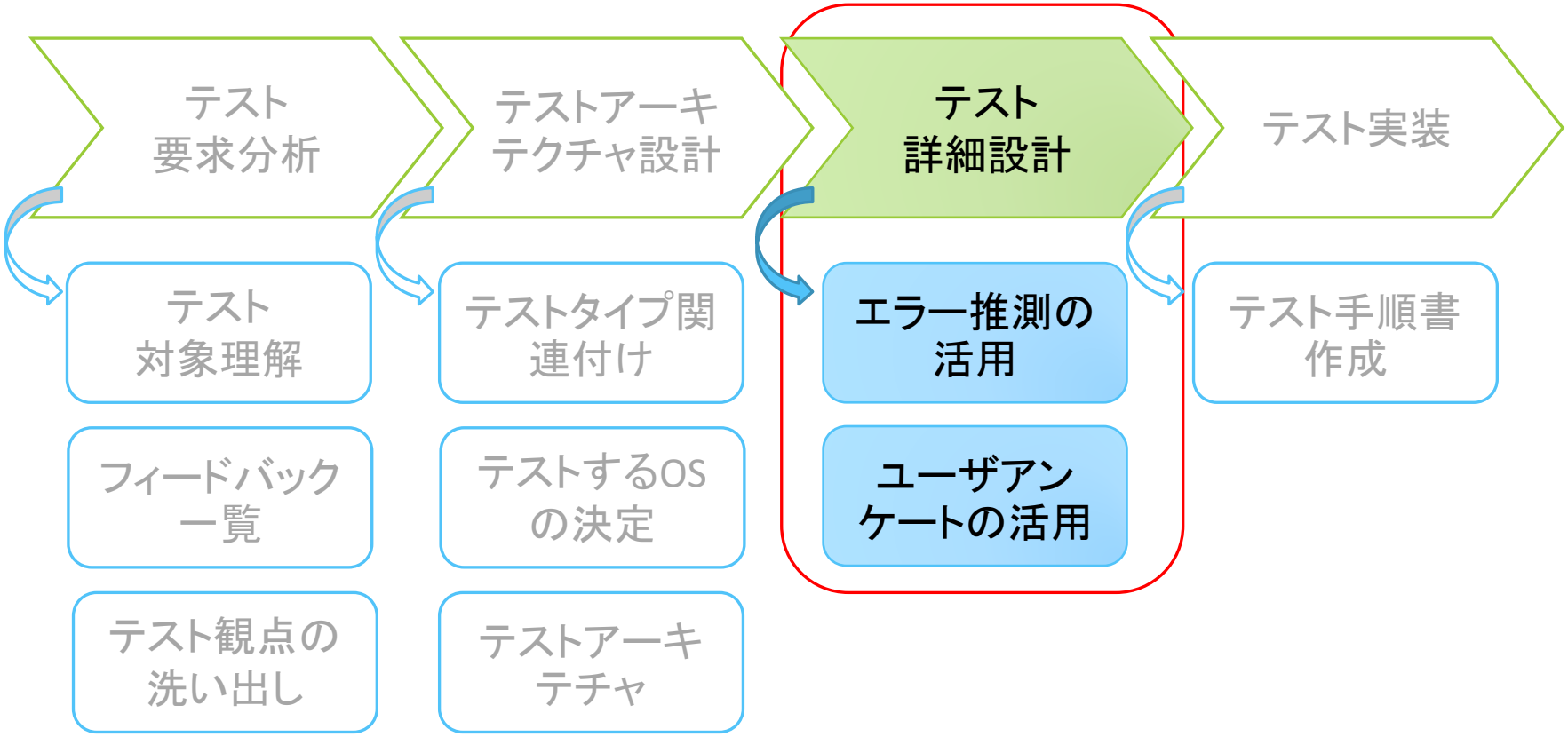


図:テスト実施計画 (一部抜粋)

3-5テスト詳細設計


テスト
要求分析
テストアーキ
テクチャ設計
テスト
詳細設計
テスト実装
テスト
対象理解
テストタイプ関
連付け
エラー推測の
活用
テスト手順書
作成
フィードバック
一覧
テストするOS
の決定
ユーザアン
ケートの活用
テスト観点の
洗い出し
テストアーキ
テチャ

エラー推測の活用

コンセプト

1.テスト実施工数に対して多くの欠陥を見つける
2.抜け漏れのないテストケース
3.割り勘の計算機能を重視したテスト設計を行う

機能テストの割り勘の複雑な計算にテスト設計技法「エラー推測」を使用

ポイント

「エラー推測」による計算間違いが起きそうな箇所のリスクの軽減

ユーザアンケートの活用

テストの目的

1.アプリケーションが用途を満たしていること

2.テストベースとソフトウェアのふるまいの合致性を検証

3.テストベースへ改善のフィードバックを行うこと

アプリの用途：2グループで支払い割合を調整する場面を想定

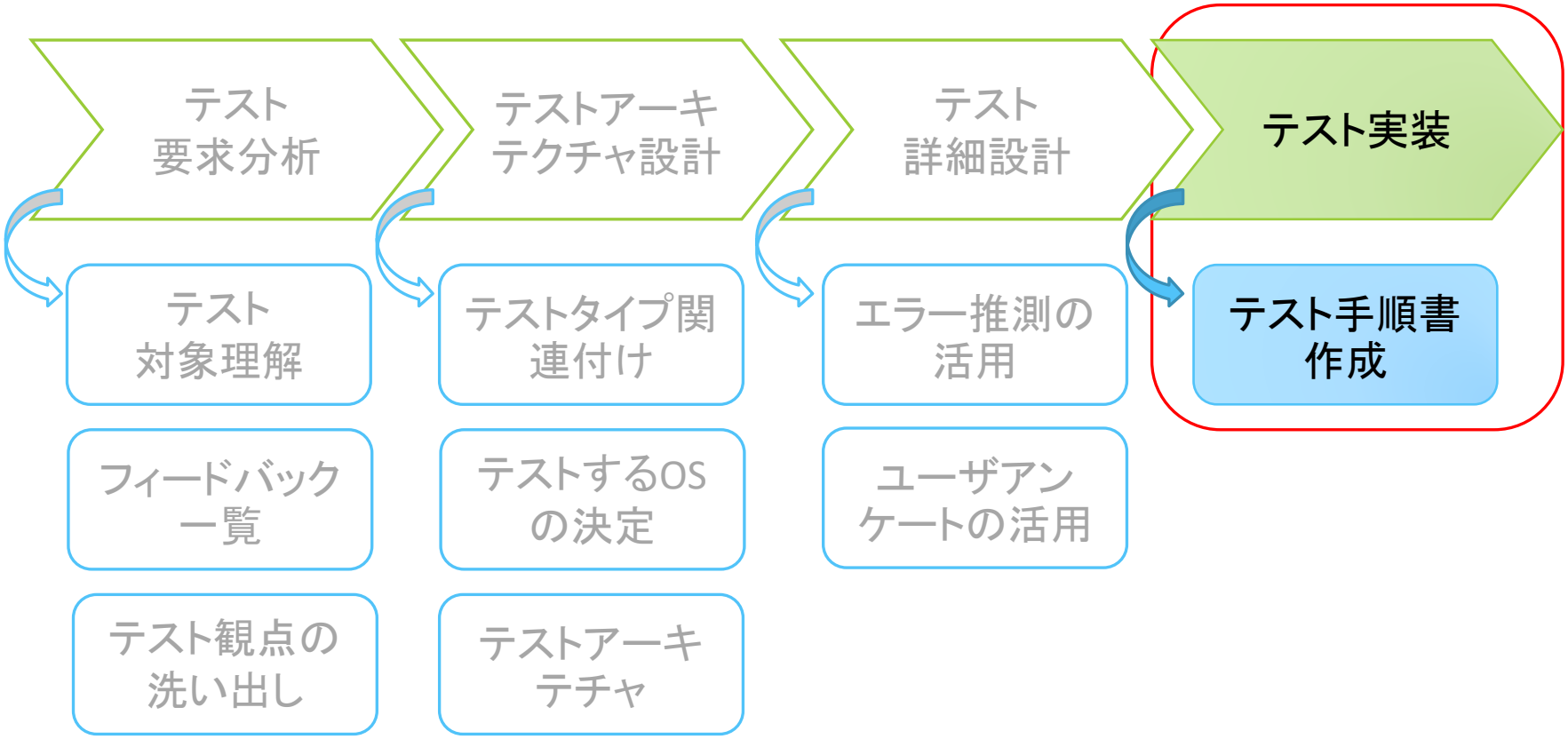
使用性テストとして使用性評価技法「ユーザアンケート」を使用

使用環境、使用状況を想定しシナリオを作成してアンケートに回答してもらう

品質副特性	Warikan テスト観点	テスト確認概要	テストで確認すること		
			上司がWarikanを利用するシナリオ	はじめてWarikanを利用するシナリオ	幹事がWarikanを利用するシナリオ
機能完全性	割り勘アプリの機能が利用者の目的を満たしているか確認する。	目的を達成するための機能は十分だったか確認する	直感的にシナリオを完遂するために必要な機能は十分でしたか	シナリオを完遂するための機能は十分でしたか	素早くシナリオを完遂するための機能は十分でしたか
機能適切性	割り勘アプリが利用用途に対し適切な仕様となっているか確認する。	目的を達成するための操作はしやすかったか確認する	シナリオを直感的に完遂するための操作はしやすかったですか	シナリオを完遂するための操作はしやすかったですか	シナリオを素早く完遂するための操作はしやすかったですか
適切度認識性	割り勘を計算できることを認識できるか確認する。	割り勘計算したことを認識できたか確認する	割り勘計算したことを直感的に認識できましたか	割り勘計算したことを認識できましたか	割り勘計算したことを素早く認識できましたか

図：使用性テスト_テスト詳細設計書(一部抜粋)

3-6テスト実装


テスト
要求分析
テストアーキ
テクチャ設計
テスト
詳細設計
テスト実装
テスト
対象理解
テストタイプ関
連付け
エラー推測の
活用
テスト手順書
作成
フィードバック
一覧
テストするOS
の決定
ユーザアン
ケートの活用
テスト観点の
洗い出し
テストアーキ
テチャ

テスト手順書作成

実施手順

1.テストケースにテスト値やテスト実施の目安時間などを追加する。
2. 使用する端末とOSを記載しテストが実施できる形に作成。
3.すべてのテストを実施したときの時間を算出する。

テストの制約では

テスト実行時間2人で2日（1日6時間相当）以内＝24時間以内

テスト手順書作成後のすべてのテスト実行時間を合わせると約20時間

テスト実行時間の制約内で設計できた！

4-1まとめ1

コンセプト

1.テスト実施工数に対して多くの欠陥を見つける
→テストタイプ毎に整理してテストの重複防止

2.抜け漏れのないテストケース
→品質特性の利用、テスト設計技法を活用

3.割り勘の計算機能を重視したテスト設計を行う
→「エラー推測」による計算間違いが起きそうな箇所のリスクの軽減

4-1まとめ2

テストの目的

1.アプリケーションが用途を満たしていること

→ユーザアンケートを利用した使用性テストの実施

2.テストベースとソフトウェアのふるまいの合致性を検証

→テストベースからテスト対象機能一覧を作成

3.テストベースへ改善のフィードバックを行うこと

→フィードバックを分類、整理し一覧を提示

ご清聴ありがとうございました