たなばた

テスト設計のご提案

2023/09/23

チーム　たなばた

チーム紹介

- チームメンバーの経験
  - データ分析、システム開発、（ちょっぴりテスト）
  - テスト経験が１年未満のメンバーで構成されています。

システムエンジニア
ひがしおか

見習いQAエンジニア
かわむら

システムエンジニア
かのう

見習いQAエンジニア
よしむら

見習いQAエンジニア
はやしだ

テスト設計のコンセプト

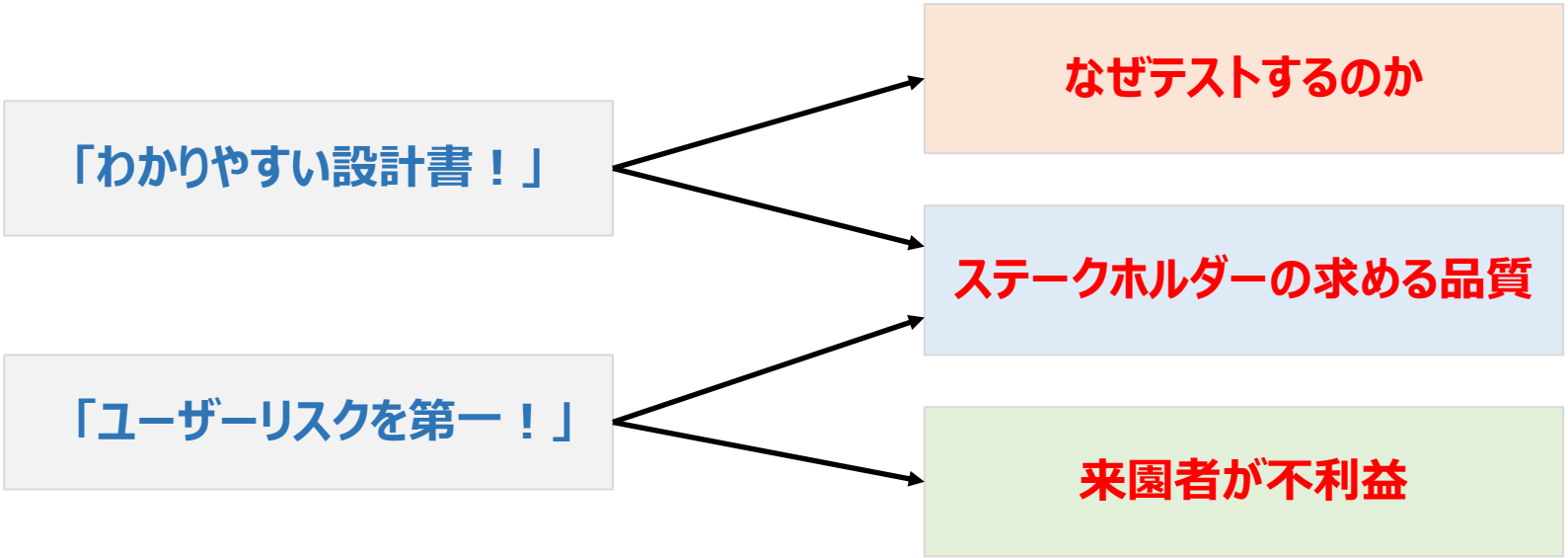
「わかりやすい設計書！」　「ユーザーリスクを第一！」

ITに詳しくない方々もテスト設計に巻き込めるようにするため、「シンプルでわかりやすい設計書」を目指す。

安全、安心をモットーに、「市場で顕在化する不具合によるユーザー影響のリスクを第一に考える」ことを目指す。

テスト設計の工夫点

- 確認すべき内容（テスト観点）を**なぜテストするのか**、明確にする。
- 依頼書の要求事項（テスト重点項目）だけでなく、各ステークホルダーの要求を見直し、**ステークホルダーの求める品質**を考える。
- **来園者（ユーザー）が不利益**を受ける不具合を重要視する。



目次

1. テスト対象について
2. テスト設計の方針
3. 要求分析
4. テストアーキテクチャ設計
5. テストケース設計
6. まとめ

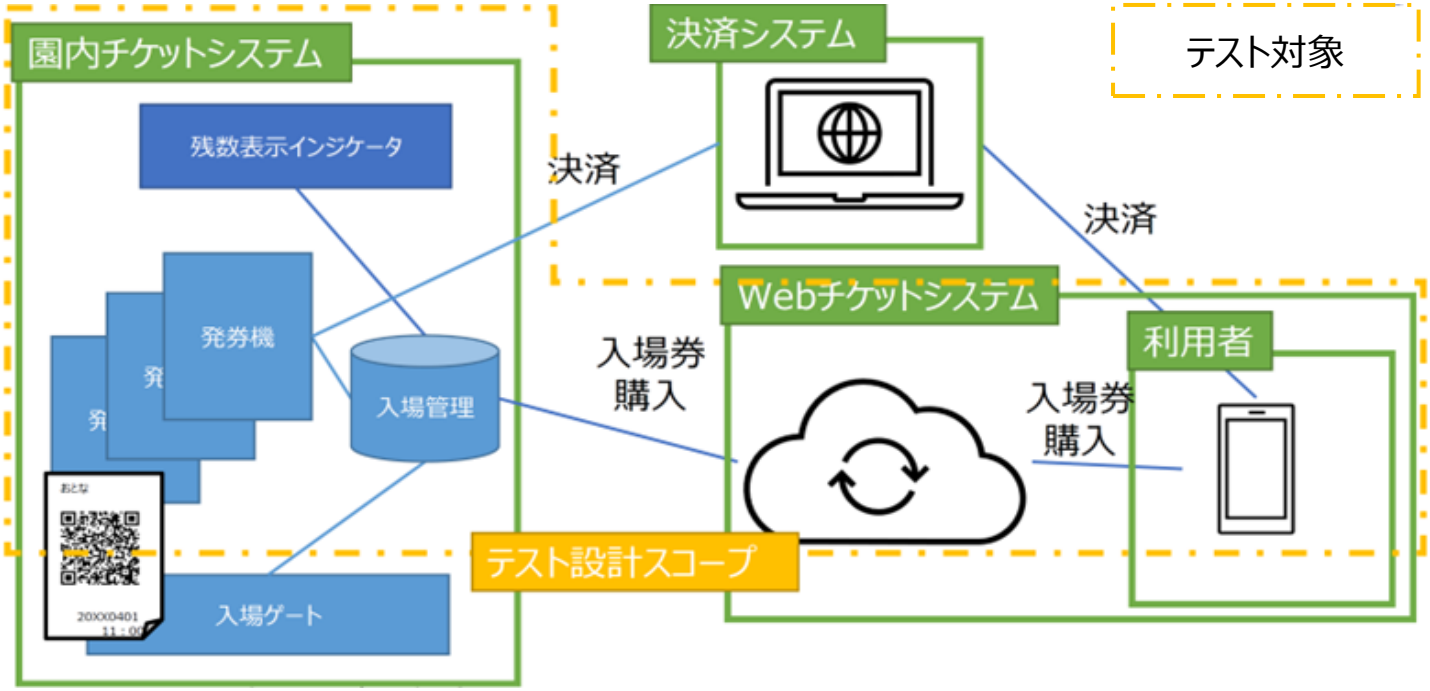
テスト対象について

■ テスト対象

だんだん入場管理システム

■ テスト範囲

- 園内チケットシステム、及びWebチケットシステム
- **<u>入場ゲート、及び決済システムはテスト対象外</u>**とする。



だんだん動物園入場システムのシステムイメージ

テスト対象について

■ 前提条件

- **<u>各サブシステム単独で</u>**確認可能な要求事項については、**<u>開発会社がそれぞれテストは実施済み</u>**である。
  - 会員登録・変更・退会、お問い合わせ、画面デザイン、QRコードの作成は保証済み。
- 既存の入場システムから**<u>差分のない箇所</u>**については、現在までの運用実績から、相応の**<u>信頼性は確保されている</u>**と考えている。
- だんだん動物園から共有されたテスト依頼書等を前提として、テスト設計を実施する。

■ 依頼者のテスト要求

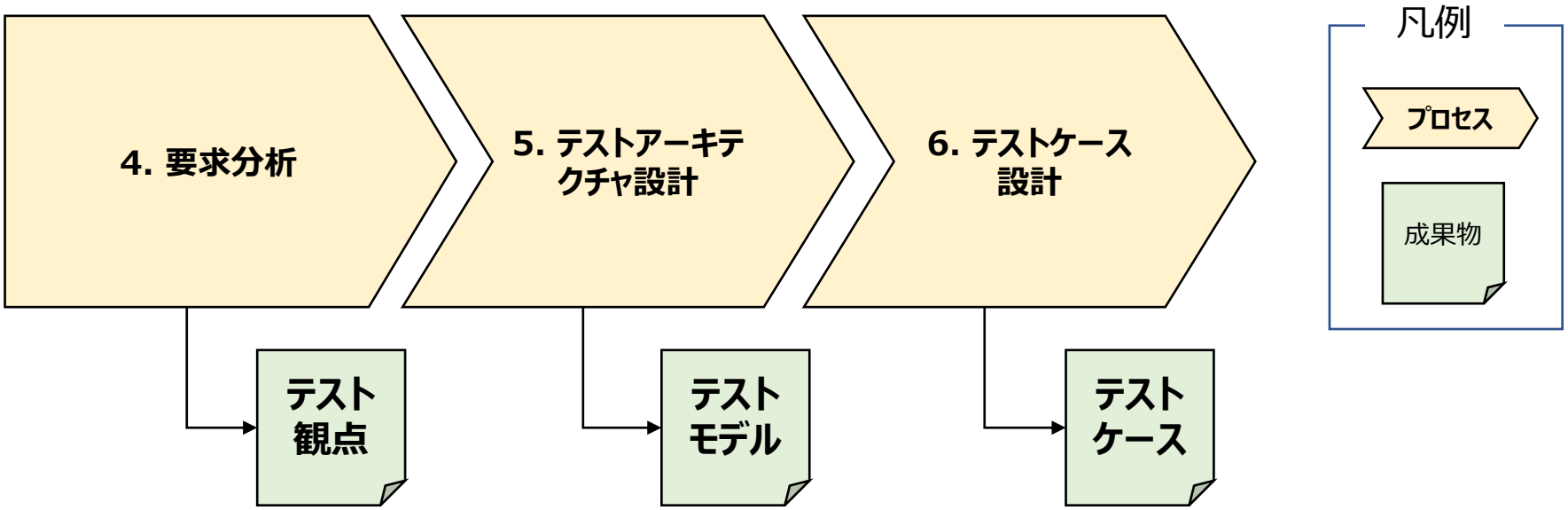
- システムが長時間稼働することを保証したい。
- 一度に制限人数以上の来場者が来てしまう事態は回避したいために時間枠の重複販売が発生しないことを保証したい。
- 今回の施策により、確実に「密回避」が達成できたことの効果を示したい
  ただし、これは既存の仕組み（入場ゲートでの入場記録と時間別入場者数の集計機能）により既に達成している。

テスト設計の方針

■ テスト設計のプロセス

注）以後、プロセス番号はテスト設計書の章番号に対応

- ３ステップでテスト設計を段階的に詳細化していく。



プロセス名	目的
4. 要求分析	満たすべき品質を特定し、**確認すべき内容（テスト観点）**の洗い出しを行う。
5. テストアーキテクチャ設計	テスト観点に対しどのようなテストを実施するか、テストの分類（テストタイプ）を検討するとともに、テスト実施の最適化を行うため、**テスト全体の構造（テストモデル）を明らかにする**。
6. テストケース設計	テストモデルで整理したテスト観点を**テスト実施可能な状態（テストケース）**にする。

要求分析

■ 要求分析の進め方

- ユーザーの要求から満たすべき品質を特定し、**テスト観点**を作成する。

進め方	目的
4.2 ステークホルダーの要求の抽出	満たすべき品質を特定する為に**ステークホルダーの要求を明らかにする**。
4.3 ユーザーストーリーの作成	要求に対するリスクを明らかにする為、ユーザーストーリーを作成する。
4.4 ユーザーストーリーを用いたリスク分析	ユーザーストーリーに対するリスク分析を行い、**要求に対するリスクを特定**する。
4.5 リスクレベル評価	リスクレベルを定義してリスク優先度を決めることで、**満たすべき品質の特定**をする。
4.6 テスト観点の抽出	リスクレベルの大きさから**テスト観点を作成する**。

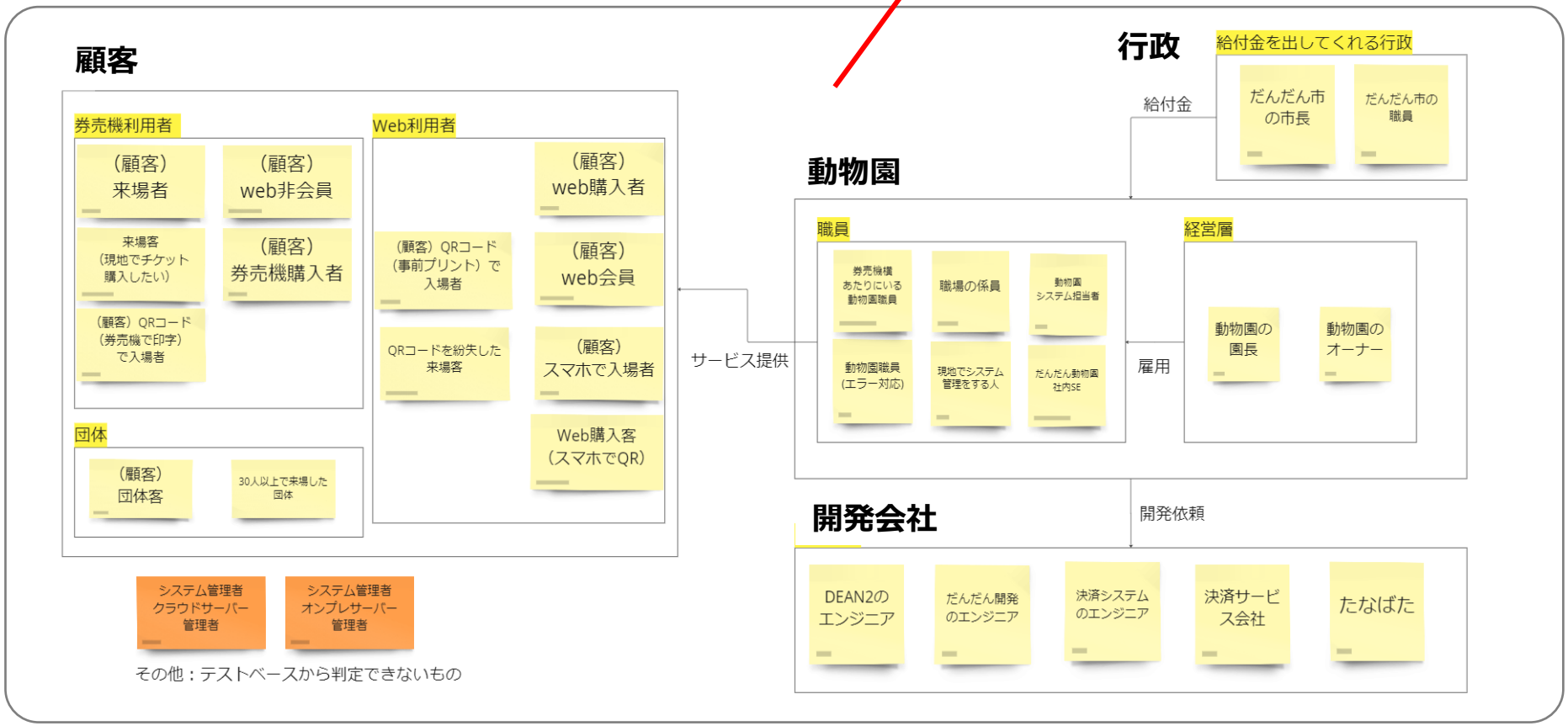
要求分析

■ ステークホルダーの要求の抽出(1/2)

- テスト依頼書から想定されるステークホルダーを整理し、**ステークホルダー4つの属性に分類した。**

ポイント

ブレスト形式で行うことで、色んな視点でステークホルダーを展開



要求分析

■ ステークホルダーの要求の抽出(2/2)

- 依頼書内容や各ステークホルダーの役割かステークホルダーが**求めるている要求を明らかにした。**

ポイント

ステークホルダーごとに担当を決めて、役割になりきって抽出した。

ポイント

5W1Hの思考整理術でステークホルダーの要求を抽出

ステークホルダー who	何に対して what	要求 how	理由 why	いつ when	どこで where
行政	動物園	感染症のクラスタを発生させないで欲しい	市民の健康を守りたいから	感染症流行時	動物園来園時に
	動物園の開発したシステム	3密回避できるシステムであることを確認したい	補助金を給付するに値するかの判断をする為に	補助金申請の 受諾時	オフィスで
	市民	市民に安心して動物園を利用して欲しい	娯楽を楽しんで欲しいから	感染症流行時	だんだん市で
動物園	Webチケットシステム 、券売機	お客さんに迷惑が掛からないよう＆園内が密にならないように稼働して欲しい	お客さんの信用を守りたいから、安心安全な施設とアピールしたい	24時間	園内で
	たなばた	「追加・変更点に品質上問題ないこと」が示された開発完了報告書が欲しい	できれば、 だんだん市から補助金が 欲しいから	だんだん市提出までに	―
顧客	発券機	購入した枚数だけ QRコードほしい 正しい料金が請求される おつりが過不足なく出てくる 購入した希望時間帯に入園できること	チケットが正しく使えないと困るから	動物園来た時	動物園前
	発券機	字が大きくてわかりやすいこと 操作が簡単 なるべく早くチケット発行してほしい	簡単にチケット購入したいから	動物園来た時	動物園前

要求分析

■ ユーザーストーリーの作成(1/2)

- 具体的なユーザーストーリー作成するため、**各ステークホルダー別にペルソナを作成した。**

ポイント

抽出した要求情報をインプットとして、作成する事で、要求に密接に関係のあるペルソナを作成

家族連れの父(Web予約)

家族連れ（父親）

年齢：４０代
性別：男性
職業：会社員
家族構成：妻、子供（６～８歳）
性格：心配性、機械音痴、老眼
価値観:嫁さん第一
余暇の過ごし方：家族サービス
人間関係：同僚、町内会、学校関係
情報源の種類：新聞、ニュース、所持しているPC、モバイル端末：スマホ

動物園の職員

職員（正社員）
年齢：３０代
性別：女性
職業：事務職員
家族構成：夫、子供（4歳）あり
性格：おだやか、仕事テキパキ
価値観:ワークライフバランス
余暇の過ごし方：友達とランチ、子どもと遊ぶ
人間関係：学生時代の友人と今も連絡取り合ってる
情報源の種類：SNS（インスタ）ウェブニュース
所持しているPC、モバイル端末：スマホ

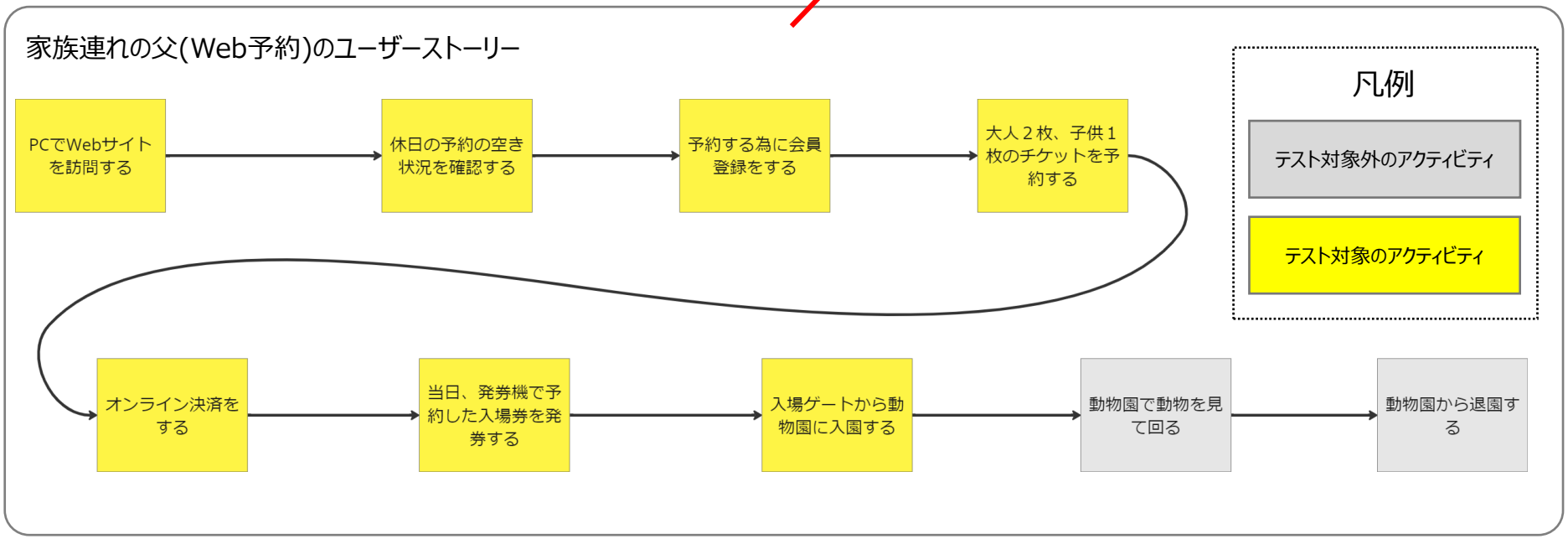
要求分析

■ ユーザーストーリーの作成

- ペルソナごとにだんだん動物園に関連する行動を整理し、**テスト対象となるユーザー行動を明らかにした。**

ポイント

ペルソナの行動を明らかにする事で、ユーザー目線でテスト観点抽出を可能にした。



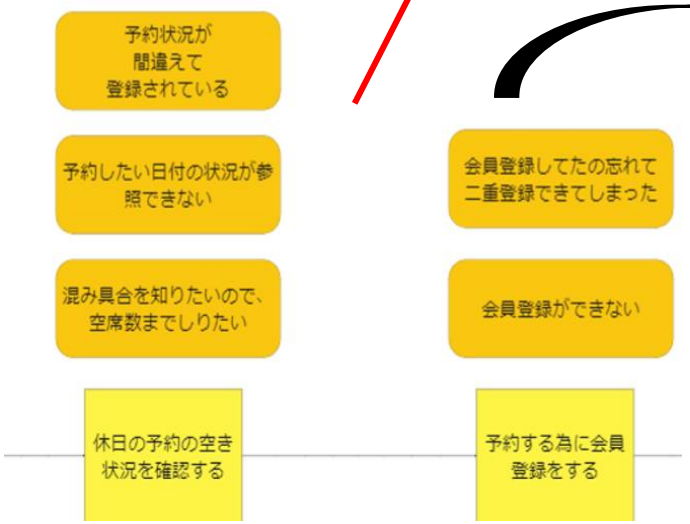
要求分析

■ ユーザーストーリーを用いたリスク分析

- ユーザーストーリーの行動に対して、発生するリスクの抽出を行った。
- また、リスクを業務内容で整理することで、抜け漏れた観点の洗い出しを行い、**ステークホルダーに対するリスクを明らかにした。**

ポイント

ブレスト形式で行動に対して発生するリスクを展開



ポイント

業務内容で整理し、全体俯瞰して観点の抜け漏れを防止

業務分類	業務詳細	アクティビティ	リスクID	リスク
事前行動	Webサイトにアクセスする	PCでWebサイトを訪問する	CF001	webサイトが閲覧できない
事前行動	Webサイトにアクセスする	PCでWebサイトを訪問する	CF002	日曜日分の販売開始日にアクセス集中でサーバが落ちる
事前行動	Webサイトにアクセスする	PCでWebサイトを訪問する	CF003	予約画面への行き方がわからない
事前行動	空き状況を確認する	休日の予約の空き状況を確認する	CF004	予約したい日付の状況が参照できない
事前行動	空き状況を確認する	休日の予約の空き状況を確認	CF005	混み具合を知りたいので、空席数ま

家族連れの父(Web予約)のユーザーストーリーに対するリスク分析

要求分析

■ リスクレベルの評価
満たすべき品質を特定する為に、**リスクレベルの指標作成**をした。

ポイント
お客様への影響の大きさと品質保証の度合いからリスクレベルを定義

ユーザーの安心度 × 品質レベル ＝ リスクレベル

ポイント
評価基準を設定することで客観的にリスクの大きさを評価

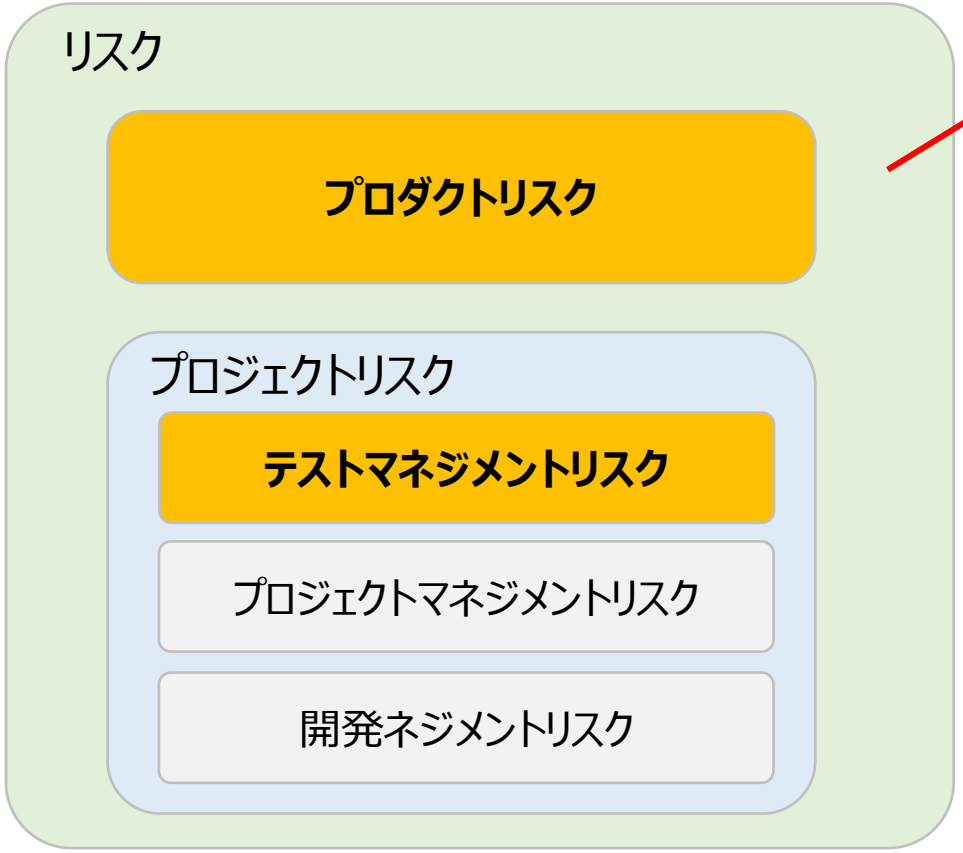
	リスクの大きさ	評価基準
ユーザーの安心度	1	3密回避できる。また、お客様に対して不利益※1が発生しない。
	2	3密回避できる。しかし、お客様に対して不利益※1が発生する。
	3	3密回避できない。
品質レベル	1	サブシステムとして保障されているかつ運用実績がある。
	2	サブシステムとして保障されているが、運用実績がない。
	3	品質保証されていない。

※1　不利益：ユーザーに何かしらの影響を与えるもの

要求分析

■ リスクレベルの評価

- リスクタイプの分類を行い、**プロダクトリスクのみをテスト対象**とした。
- プロジェクトリスクは、更にリスクタイプの分類を行い、**テストマネジメントリスクのみをスコープとして、対応策を検討**した。



ポイント

たなばたがコントロールすることが可能であるリスクのみを対象

凡例

対象　対象外

要求分析

■ テスト観点の抽出

- **サブシステム単体で発生しうるリスク**は、開発会社でテスト実施済みのため**テスト対象から除外した。**
- **リスクレベルが6以上のリスクを**満たすべき品質の基準として、**テスト観点に設定した。**

ポイント

品質が担保されておらず、ユーザーに影響を与えるリスクであるリスクレベル6以上を対象とした。

リスクID	リスク	リスクタイプ	サブシステム間で発生するリスク	顧客の安心度	品質レベル	リスクレベル
CF007	購入できないのに、購入できると表示される	プロダクトリスク	○	3	3	9
ZW004	実際は購入できるが、残数枠数が０になっている	プロダクトリスク	○	2	3	6
ZW010	エラー発生してないのに エラー表示が出る	プロダクトリスク	○	2	3	6
ZW001	電源ONできない	プロダクトリスク	―	2	2	4
CF001	webサイトが閲覧できない	プロダクトリスク	―	2	2	4
CC004	購入可能枠はないが、購入できる	プロダクトリスク	―	1	2	2
GM001	審査基準がわからない	プロジェクトリスク	―	―	―	―
ZD001	給付金の期限は上限はないか	プロジェクトリスク	―	―	―	―

対象

対象外

リスクレベルからテスト観点の優先順位付け

アーキテクチャ設計

■ アーキテクチャ設計の進め方

- テスト観点の関連システムとテストタイプを明らかにすることで、**テストモデル**を作成する。

進め方	目的
5.2 テスト観点の詳細化	テスト観点を詳細化することで、**確認すべき内容の具体化**を実施する。
5.3 テストタイプの整理	各テスト観点に対して、**確認すべき内容のテスト種類の分類（テストタイプ）**をした。
5.4 テストモデルの作成	テスト観点が関連するシステムとテストタイプを整理することで、**テストモデルを作成した。**

アーキテクチャ設計

■ テスト観点の詳細化

- テスト観点に対して、FTAを用いることで不具合の原因から**具体的なテスト観点**を作成した。

ポイント

FTAにより不具合の原因を分析することでテスト観点の具体化を実施

リスクID（CF007)のテスト観点ツリー

テスト観点

購入できないのに、購入できると表示される

不具合の原因	詳細化したテスト観点
通信遅延	Webチケットシステムへのアクセス数増加による通信負荷
通信障害	Webチケットシステムへのアクセス数増加による通信負荷
	webチケットシステムで通信障害時にエラー処理がされない
DB改ざん	WebチケットシステムのDBのアクセス権が漏洩する
時刻同期	Webチケットシステムと入場管理システム間の同期ができていない
実装ミス	webチケットシステムと入場管理システム間の通信
設定ミス	入場管理管理システム予約可能枠の上限設定の誤りがある
	Webチケットシステム予約可能枠の上限設定の誤りがある

アーキテクチャ設計

■ テストタイプの整理

- 詳細化したテスト観点ごとに**テストタイプを分類した。**

ポイント

テスト観点ごとに、テストタイプを設定することで
テストの構造化をしやすくする。

テスト観点ID	テスト観点詳細	テストタイプ
CF007-1	Webチケットシステムへのアクセス数増加による通信負荷	ストレステスト
CF007-2	webチケットシステムで通信障害時にエラー処理がされない	障害許容性テスト
CF007-3	券売機で通信障害時にエラー処理がされない	障害許容性テスト
CF007-4	WebチケットシステムのDBのアクセス権が漏洩する	セキュリティテスト
CF007-5	Webチケットシステムと入場管理システム間の時刻同期ができていない	互換性テスト
CF007-6	webチケットシステムと入場管理システム間の通信の実装ミス	機能確認テスト
CF007-7	入場管理管理システム予約可能枠の上限設定の誤りがある	機能確認テスト
CF007-8	Webチケットシステム予約可能枠の上限設定の誤りがある	機能確認テスト

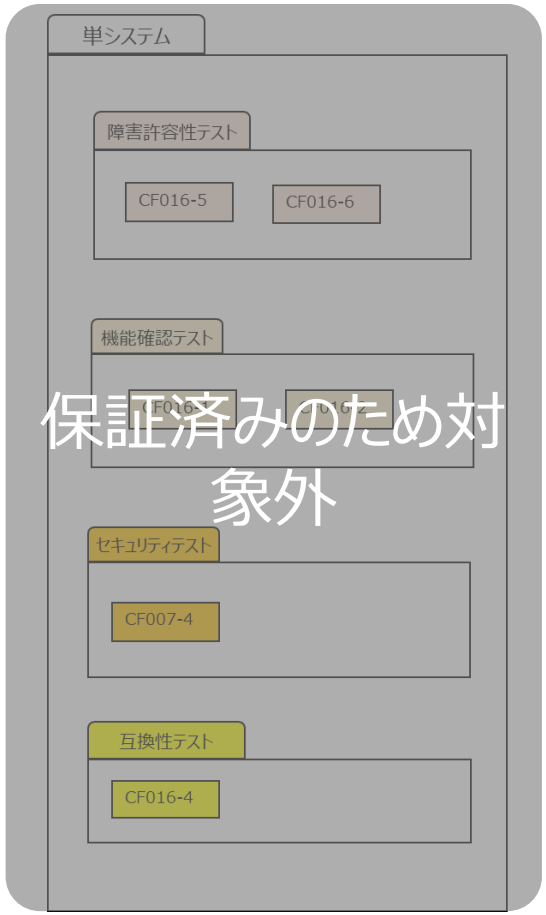
アーキテクチャ設計

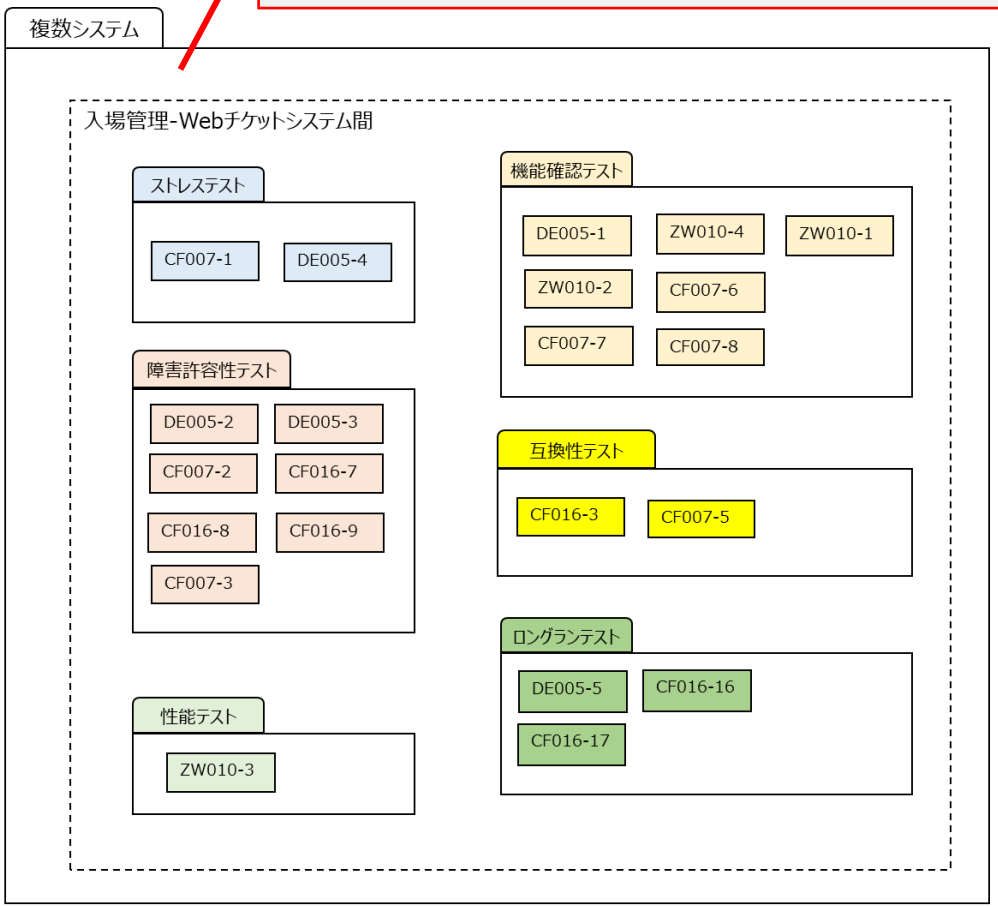
■ テストモデルの作成

- テスト観点が関連するシステムとテストタイプを整理することで、**テストモデルを明らかにした。**

ポイント

関連システム、テストタイプからテストモデルを作成することで、テスト実施の順序を整理する。





テストケース設計

■ テストケース設計の進め方

- 各テスト観点に対して実施する、テスト条件を明らかにすることで**テストケース**を作成する。

進め方	目的
6.2 テスト条件の選定	各テスト観点に対して、どのような条件でテストを実施するか**テスト条件を定める**。
6.3 テストケースの作成	テスト条件からテスト実施の方法を具体化することで、**テストケースを作成する**。

テストケース設計

■ テスト条件の選定

- テスト観点をテスト技法により整理し、**テストに必要な条件を整理した。**

■ テストケース作成

- 上記の内容から、整理したテスト条件から**具体的なテストの実施手順を作成して、テストケースを作成した。**

ポイント

利用ユーザーを明確にしたテストケース作成した。

テストタイプ	テスト観点ID	テスト条件	対象ユーザー	実施手順	期待結果
ロングランテスト	CF016-16	【前提条件】 ・Webチケットシステムの予約管理DBのレコード数が、一ヶ月の想定来場者数を超えているとき ・会員登録しているユーザーであること ・事前にWebチケットシステムで予約していること 【事前準備】 ・一ヶ月の想定来場者数のレコードを事前にwebチケットシステムのDBに登録しておく。 ・Webチケットシステムで予約購入しておく。	券売機での発券する人	1. 発券機でログインする。 2. 予約したチケット発券する。	発券できることを確認する。
機能確認テスト	CF007-6	【前提条件】 ・Webチケットで時間枠の予約の残数が0のとき ・残数0にした後、90分後に実施する。 【事前準備】 ・Webチケットシステムの予約管理DBの予約枠を残数0にする。	発券機で購入する人	1. 発券機で今すぐ入場券を選択する。	今すぐ予約購入が選択できない

補足：テスト条件の展開例（テスト観点ID:CF007-6）

■ テスト条件の展開例

- テスト観点詳細から関連するシステムと機能をもとに、仕様書を確認する。
- テスト観点ID:CF007-6のテスト条件を整理した図を以下に示す。

<table>
<tr><th colspan="2" rowspan="2"></th><th colspan="7">時間枠</th></tr>
<tr><th>~30分以内</th><th>30分以降～60分以内</th><th>60分以降～90分以内</th><th>・・・</th><th>17:00~17:30</th><th>・・・</th><th>24時間以降~30日後まで</th></tr>
<tr><td rowspan="2">発券機</td><td>今すぐ購入</td><td>○</td><td>×</td><td>×</td><td>×</td><td>×</td><td>×</td><td>×</td></tr>
<tr><td>時間指定購入</td><td>×</td><td>○</td><td>○</td><td>○</td><td>○</td><td>×</td><td>×</td></tr>
<tr><td>webチケットシステム</td><td>予約購入</td><td>×</td><td>×</td><td>×</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td colspan="2" rowspan="2">データの流れ</td><td colspan="3">予約管理DB→入場管理システム</td><td colspan="2">webチケットシステム→予約管理DB</td><td colspan="2" rowspan="2">webチケットシステム→予約管理DB</td></tr>
<tr><td colspan="3">券売機→入場管理システム</td><td colspan="2">券売機→入場管理システム→予約管理DB</td></tr>
<tr><td colspan="2">データの管理場所</td><td colspan="3">入場管理システム</td><td colspan="4">予約管理DB</td></tr>
</table>

まとめ

- 確認すべき内容（テスト観点）を**なぜテストするのか**、明確にする。

要求とテスト観点が紐づいたテスト設計を行うことで、テストの意図を把握できるようにした。

- 依頼書の要求事項（テスト重点項目）だけでなく、各ステークホルダーの要求を見直し、**ステークホルダーの求める品質**を考える。

ステークホルダーのロールになりきった要求分析とペルソナの作成により、ステークホルダーの求める品質を抽出することができた。

- **来園者（ユーザー）が不利益**を受ける不具合を重要視する。

ユーザー安心度を用いたリスクレベルで評価することで、来園者の不利益を重視したテスト設計を実施することができた。