

テスト設計コンテスト' 26
OPENクラス 予選プレゼンテーション資料
だんだん動物園入場システムV03テスト設計

しち



アジェンダ



1. はじめに
2. テスト要求分析
3. テストアーキテクチャ
4. テスト設計
5. テスト実装
6. まとめ

はじめに



- 自己紹介

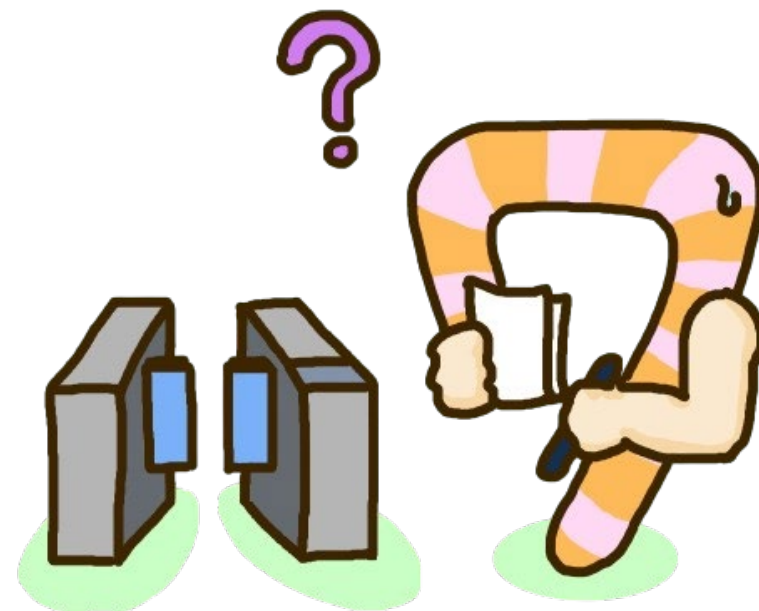
名前: しち

(由来はありません)

職業: 普通のテスター

テスト設計コンテストは

4年ぶり2回目の出場





はじめに

全てのテスト作成完了までは、できませんでした。

AIに頼り、テストの粒度が揃わず悩みました。

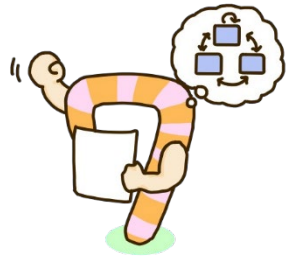
悩みながらも**過剰な作りこみを防ぎ、何をテストし、何をテストしないか**を考えました。

前回出場時は要求分析の途中で終わりました。

自分も含む未来のテスト設計コンテスト参加者が前に進む布石として、発表します。

テスト要求分析





テスト要求分析

要求の元となる情報3点

- ・今回の変更点(システム)
- ・今回の変更点(社会的な観点)
- ・強くテストをしてほしい点



テスト要求分析

要求の元①から導かれる確認すべきこと

今回の変更点(システム)

- ・入場ゲートハブ追加→経路が増えることによる、同期ミス・通信ラグ
- ・ゲート増設→同時処理
- ・残数のパラメータ処理→残数上限可変処理に柔軟対応



テスト要求分析

要求の元②から導かれる確認すべきこと

今回の変更点(社会的な観点)

- ・入場人数緩和
- ・人気動物で入場者増加
- ・近隣マンション建設(当日券の増加)

→人数制御をより正確にすることが求められる

→人数が増えるので同時・並列処理、負荷処理を確認する必要がある



テスト要求分析

要求の元③

強くテストをしてほしい点

- ・入場ゲートの変更点を考慮したリグレッションテスト
- ・今後変更時に大きな問題とならないようリグレッションテスト

→今回の変更点、将来変更する可能性がある点のリグレッションを考慮

重点項目

- ・システムが長時間稼働することを保証→長時間稼働と縮退運転確認
- ・制限人数以上の来場回避(重複販売回避)→残数制御・重複販売回避



テスト要求分析

テスト要求を考えるためのキーワード:これらがキーワードとなる

- ・入場人数・残数制御
- ・残数上限可変処理
- ・同時処理
- ・通信不安定
- ・負荷・長時間稼働

→後述のテストアーキテクチャにて、これらのキーワードにテストが割り当たっているかを確認する

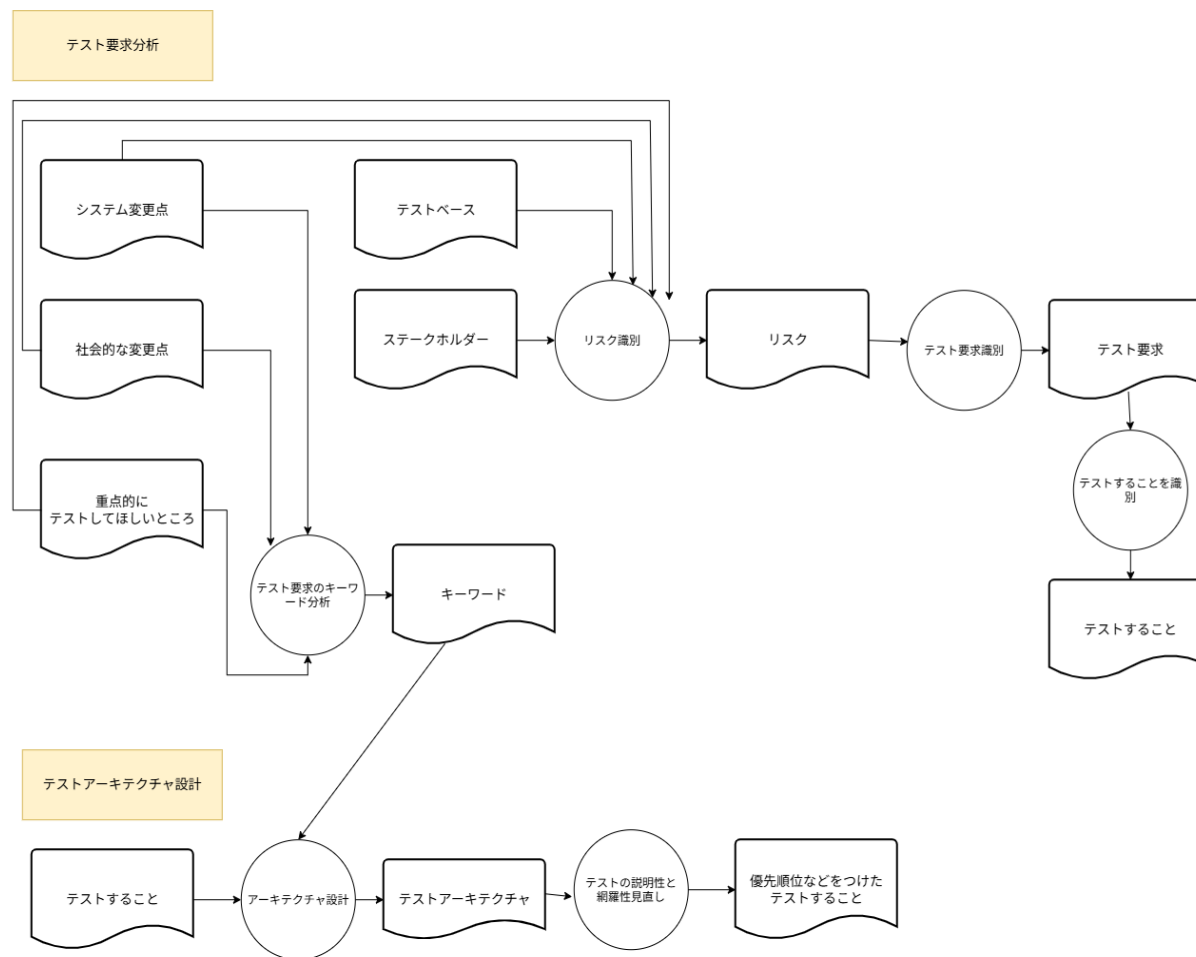


テスト要求分析

テスト要求

- ・変更点・仕様・依頼事項から重点箇所を整理し、AIでリスク候補を洗い出した後、人間が精査する
- ・精査したリスクや変更点・重点確認事項を元に「テスト要求は何か」「テスト要求のために行うテストは何か」を考えていく

テスト要求分析



テスト要求分析



リスク一覧

目的：起きてほしくないことを考えたうえでテストでリスクを軽減できるか・今回の変更点に関係するかを考慮し、テスト要求の源泉とする

1. リスク一覧

リスクID	概要	リスク内容	想定される損害	テストでリスクを軽減できるか？	今回のシステム変更点に関係するか？	今回の社会的変更点に関係するか？	今回変更点に関係あるか：詳細	テスト範囲とするか否か
RSK-01	販売数不整合	販売経路間の残数・上限管理の不整合により、販売可能数を超過して入場券が発行される	入場者数を制御できず、入場拒否、ゲート前滞留、園内混雑、返金・窓口対応が発生する	○	○	×	残数・販売上限のパラメータ化およびゲート増設により、販売数と残数の不整合が発生する可能性がある	対象とする
RSK-02	残数管理異常	残数の同期不良や誤計算により、実際には残数があるのに販売不可になる	来園希望者が購入・入場できず、機会損失や問い合わせ・苦情が増える	○	○	×	入場関連(ゲート・ゲートハブ・入場管理)の通信で不整合が出ると発生する	対象とする
RSK-03	通信断時不安全制御	障害・通信断・タイムアウト時に判定不能でもゲートが開き、本来入れない人が入場する	人数制御が崩れ、不正入場や安全上限超過が発生する	○	○	×	入場関連(ゲート・ゲートハブ・入場管理)の通信で不整合が出ると発生する	対象とする
RSK-04	二重入場	使用済み更新の遅延・欠落や競合により、同一QRコードで複数回・複数ゲートから入場できる	不正入場、実入場者数の過少把握、残数・利用実績の不整合が発生する	○	○	×	入場関連(ゲート・ゲートハブ・入場管理)の通信で不整合が出ると発生する	対象とする
RSK-05	同時入場判定失敗	複数ゲートの同時アクセスで判定結果が混雑し、別ゲート・別QRの応答が取り違えられる	本来入れる人が入れず、本来入れない人が入る可能性があり、ゲート前対応も増える	○	○	×	入場関連(ゲート・ゲートハブ・入場管理)の通信で不整合が出ると発生する	対象とする

AIで出した結果を人間の目で妥当か判断をしていく

テスト要求一覧

目的：リスクや今回の変更点・重点確認事項から導き出したテスト要求について、確認理由およびどのテストで確認するかを明確にする

テスト要求ID	関連リスクID	テスト要求	確認理由	関連テスト	備考
TR-01	RSK-01	園内およびWebの各販売経路で、正常に予約・購入できること	システム変更後に、来園者が既存の購入経路利用でチケットを取得できないことを防ぐため	TS-01：発券機・Web同時購入テスト TS-04：各販売経路で通常通りの購入ができるかのテスト TS-17：Webチケットの同時購入テスト TS-20：発券機の同時購入テスト	
TR-02	RSK-06	発券から入場までの一連の流れを正常に実施できること	発券と入場の連携不備により、購入済みの来園者が入場できないことを防ぐため	TS-10：入場者のシナリオテスト（発券～入場） TS-12：システム起動のシナリオテスト（起動～入場開始）	
TR-03	RSK-05	入場ゲートを介した入場可否処理が正常に行われること	誤入場または有効な来園者の入場拒否を防ぎ、システム変更後も入場機能の根幹機能を維持するため	TS-06：通信不安定状態時のテスト TS-07a：QR使用テスト(機能) TS-07b：QR使用テスト(実機) TS-09：ゲート同時通過テスト TS-10：入場者のシナリオテスト（発券～入場） TS-11：時間枠切り替え境界でのゲート通過テスト TS-12：システム起動のシナリオテスト（起動～入場開始） TS-16：様々なパターンでのゲート通過テスト TS-22：ゲートハブ起動時/入場券情報一覧取得・保持テスト	

精査したリスクや変更点・重点確認事項から要求を導き、要求からテストすることを導く

テストの種類一覧

目的：テストすることの一例と要求の関連性を明確化する。また、実機で確認するべき内容が否かを明確化し、テスト実行順の考慮材料とする

ID	テストの種類	割付	テストで確認したいこと	主要求	関連要求	実機・シミュレーター	実機・シミュレーターの理由	備考	テストする
TS-01	発券機・Web同時購入整合テスト	3：同時処理	・複数の販売経路から同時に購入しても、残数が正しく更新されること	TR-04 残数・販売上限管理	TR-01、TR-06	実機	同時処理制御がメインの内容	システム的には今回の変更点ではない ただし社会的要因に関連する	テストする（探索的テスト）
TS-02	各販売経路の販売上限付近の境界値テスト	1：入場人数・残数制御	・複数の販売経路の販売上限で残数超過して販売しないこと	TR-04 残数・販売上限管理	TR-06	シミュレーター	機能確認であり処理タイミングを伴わないためシミュレーターで代用可能	システム的には今回の変更点ではない ただし社会的要因に関連する	テストする（探索的テスト）
TS-03	各時間枠販売上限付近の境界値テスト	2：残数上限可変処理	・各時間枠の販売上限を超えてチケットを販売しないこと	TR-06 人数上限の遵守	TR-04	シミュレーター	機能確認であり処理タイミングを伴わないためシミュレーターで代用可能		テストする
TS-04	各販売経路で通常通り購入できるかのテスト	1：入場人数・残数制御	・複数の販売経路で通常通りチケットが購入できること	TR-01 予約・購入	TR-04	両方	スモークテストとしての意味 実際のシナリオを重視		テストする
TS-05a	残数増減テスト(機能)	1：入場人数・残数制御	・残数変化のタイミングで残数が適切に増減すること(機能のトリガー)	TR-04 残数・販売上限管理	—	シミュレーター	機能確認であり処理タイミングを伴わないためシミュレーターで代用可能		テストする
TS-05b	残数増減テスト(通信タイミング)	4：通信不安定	・残数変化のタイミングで残数が適切に増減すること(通信のタイミング)	TR-04 残数・販売上限管理	—	実機	通信制御として動作するか		テストする



テスト要求分析

テスト要求分析段階で、やりすぎを減らす①

- ・過度なステークホルダーの分類

今回の入場に関しての変更箇所は、ゲートが複数になる、処理がハブ経由になる、入場者増加などがあるが

システム側から見た場合「同時処理人数」「通過速度(次の判定のタイミングの間隔)」に集約される。

→よってステークホルダー識別はするが、過度な分類はしない

- ・過度な分類の例

サブシステム担当別の過度な分解(ゲート担当・残数表示担当など)

来園者で通過速度が変わらない分類(中学生・高校生など)



テスト要求分析

テスト要求分析段階で、やりすぎを減らす②

- ・今回の実機は「発券機」「ゲート」など、台数を多数用意できるようなものではないため、実機確認はどうしても時間がかかる
- ・テストの内容を考え、タイミングを伴うものではなく、ロジックのみを確認するものはシミュレーターで確認する

シミュレーターで確認できるもの→機能確認そのもの

通信タイミングや同期、負荷など、実機特有の考慮条件を伴わないもの



テスト要求分析

リグレーションテストで確認すべき箇所の考察と対策

・今後変化する可能性がある箇所

1. 人数(各販売時間枠の上限)

→リグレーションリスク: 人数を増やしたことが機能に反映されない、想定以上の負荷が発生する

→人数を可変パラメータにする

2. システムのリプレイスやハブ増設

→リグレーションリスク: サブシステムの入れ替えでインターフェース間のデータやり取りが意図せず変わる

→ 各サブシステムのインターフェースの確認テストを作り、インターフェース間でやり取りしているデータが変わらないことを確認する

テストアーキテクチャ





テストアーキテクチャ

テストアーキテクチャとは何をするためのものか？

→テスト全体像を構造化して、骨組みや指針を見やすくしたもの

見やすくしてどうするか？

→テスト全体を理解しやすくする

何のために？

→①関係者と判断や説明を合意しやすくするため

→②そのためにまず自分が理解するため



テストアーキテクチャ

説明・理解しやすくするために以下の3アーキテクチャを組み立て

①要求のキーワードに対しての「テストすること」

→テスト要求に対しての抜けが無いかの可視化

②今回の変更点に直接影響がある箇所とリグレッション確認範囲

→変更点に対して適切にテスト確認できているか・漏れや過剰テストが無いかの確認

③テスト実行順

→どのテストから実施するかの確認



テストアーキテクチャ

アーキテクチャ① 要求のキーワードに 対しての 「テストすること」

テストアーキテクチャ①：要求キーワードに対するテストの割り付け

グループ1：入場人数・残数制御
TS-02 各販売経路の販売上限付近の境界値テスト
TS-04 各販売経路で通常どおり購入できるかのテスト
TS-05a 残数増減テスト（機能）
TS-07a QR使用テスト（機能）
TS-10 入場者のシナリオテスト（発券～入場）
TS-11 時間枠切り替え境界でのゲート通過テスト
TS-12 システム起動のシナリオテスト（起動～入場開始）
TS-19 残数インジケータの同期テスト
TS-21 入場人数上限値設定変更・反映テスト
TS-22 ゲートハブ起動時/入場券情報一覧取得・保持テスト
TS-24 各サブシステム機能テスト

グループ2：残数上限可変処理
TS-03 各時間枠販売上限付近の境界値テスト
TS-11 時間枠切り替え境界でのゲート通過テスト
TS-12 システム起動のシナリオテスト（起動～入場開始）
TS-21 入場人数上限値設定変更・反映テスト

グループ3：同時処理
TS-01 発券機・Web同時購入テスト
TS-09 ゲート同時通過テスト
TS-16 様々なパターンでのゲート通過テスト
TS-17 Webチケットの同時購入テスト
TS-20 発券機の同時購入テスト

※色付きのテスト
（TS-11、12、16、17、20、21）
...複数の要求キーワードに割り
当たっているテスト

グループ4：通信不安定
TS-05b 残数増減テスト（通信タイミング）
TS-06 通信不安定状態時のテスト
TS-07b QR使用テスト（実機）
TS-11 時間枠切り替え境界でのゲート通過テスト
TS-12 システム起動のシナリオテスト（起動～入場開始）
TS-14 エラー検知テスト
TS-15 異常検知時の縮退運転継続性確認

グループ5：負荷・長時間稼働
TS-13 ロングランテスト
TS-16 様々なパターンでのゲート通過テスト
TS-17 Webチケットの同時購入テスト
TS-20 発券機の同時購入テスト

グループ6：インターフェース（複数のキーワードを横断して、サブシステム間のデータ連携を確認する）
TS-23 データ連携仕様適合テスト



テストアーキテクチャ

アーキテクチャ①の狙い

→各キーワードに対して確認すべきテストの抜けがないことを確認する

テストアーキテクチャ



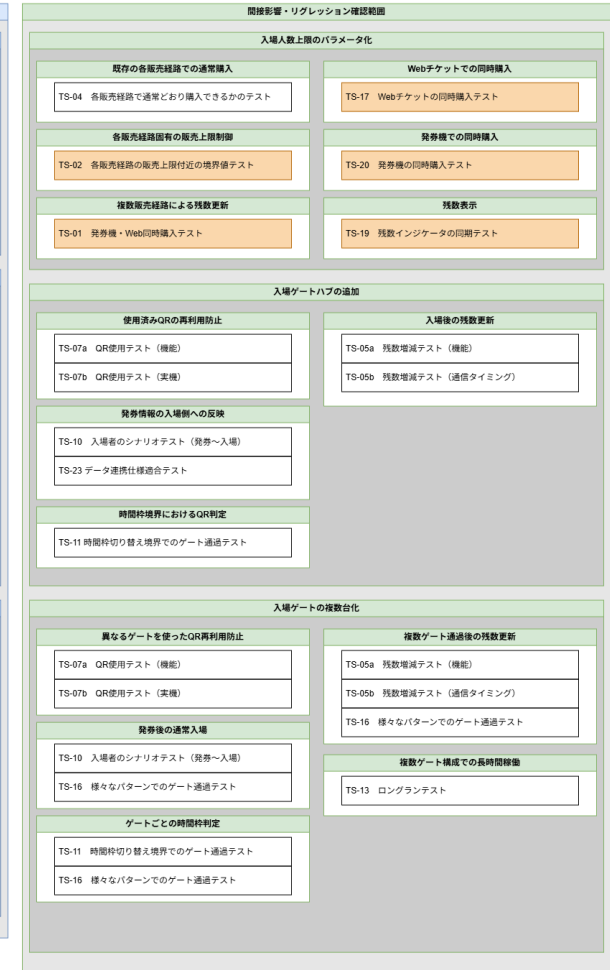
アーキテクチャ②

②今回の変更点に 直接影響がある箇所と リグレッション確認範囲

テストアーキテクチャ②：変更影響とリグレッション確認範囲



※色付きのテスト
(TS-01、02、17、18、20)
...変更点からの影響が間接的なため、代表条件を探索的に確認するテスト





テストアーキテクチャ

アーキテクチャ②の狙い

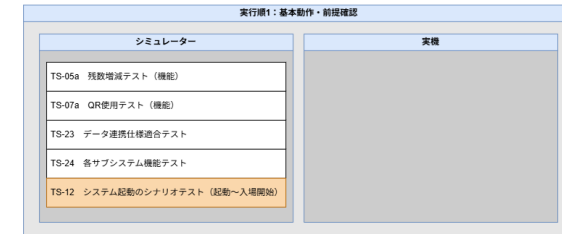
- テストする深さについてメリハリをつける
- 今回の直接の変更点か間接変更点なのかを分け、リグレッションテストに分類されるか否かを分ける

テストアーキテクチャ



アーキテクチャ③ ③テスト実行順

テストアーキテクチャ③：テスト実行順



※色付きのテスト
（TS-12、04、11、14、21、15）
...シミュレーター、実機両方で確認





テストアーキテクチャ

アーキテクチャ③の狙い

→テスト実行順を可視化し、シミュレーター／実機を含めた実施順の認識に齟齬がないか確認する



テストアーキテクチャ

アーキテクチャ俯瞰で、やりすぎを減らす

- ・購入処理そのものの過度なバリエーションテスト

理由: 購入処理は今回の変更ではない。

よって

「(前回の)入場システムから差分の無い箇所については、現在までの運用実績から、相応の信頼性は確保されていると考えている」

という記述の「差分の無い箇所」に含まれていると推測できる

→残数更新処理にゲートが加わることで同時処理の影響を受けるが、購入処理自体は変更なし

→探索的テストで同時処理を確認する



テストアーキテクチャ

テストアーキテクチャのまとめ

- ・必要なテストが揃っているかのアーキテクチャ(①)

→各カテゴリに対して「このテストで実施している」ことを分かるようにする

- ・どこをどうテストするかのアーキテクチャ(②・③)

→「重点的にテストするところ」「どこからテストするか」を分かるようにする

※各アーキテクチャともに常に「狙い」に立ち戻れるようにして「**何のためのアーキテクチャか**」を分かるようにする

テスト設計



テスト設計



テスト設計では
そのテストの目的・実際に使われるときの影響の強さを元に
カバレッジを割り出していく



テスト設計

各テストについて下記を出しておく

- ・目的
- ・発生した場合に問題となる不具合
- ・確認の優先度
- ・テスト設計に使用する情報・パラメータ類

→テストを詳細化していくときに、常に目的に戻って粒度を考えられるようにしておく

テスト設計



テストID	テスト名	目的	関連するシステム変更点			
TS-03	各時間枠販売上限付近の境界値テスト	各時間枠の販売上限を超えてチケットを販売しないことを確認する。	入場制限者数を倍(30分あたり30人→60人、パラメータで容易に変更可能0～999人)			
発生した場合に問題となる不具合	実機/シミュレーター	確認の優先度	特記事項			
上限を超えた数のチケットが販売され、想定した以上の人数が入場できてしまう	シミュレーター	高/実行順2	・可変項目のためリグレッションテスト時にパラメータ値を変更できるよう留意すること ・各パラメータ設定時のシステムの表示についてはTS-21で実施する			
テスト設計に使用する情報・パラメーター						
確認項目						
概要	確認ポイント					
販売時の境界値テスト	各枠の販売数境界値で販売可・不可の動作が正しいこと					
パラメータ1：残数枠の種類						
残数管理場所	経由先	購入方法	販売経路・対象枠	※残数の更新経路別で同値分割する		
入場管理	-	発券機	現在・次・次々枠	クラウド2が管理している残数	5-3.時間枠別残数（全体）	
クラウド2	入場管理	発券機	現在・次・次々枠より先の時間指定枠		1-3.時間別残数情報（当日分全体）	
クラウド2	クラウド1	Web	Webで購入可能な時間指定枠	入場管理が管理している残数	1-2.時間枠別残数（現在・次・次々枠分）	
パラメータ2：設定する上限値						
パラメータ	確認理由					
60	今回変更される値					
999	パラメータで可変の最大値					
0	パラメータで可変の最小値			※要仕様確認：0を有効な設定値として扱ってよいか（仕様確認後作成とする）		
パラメータ3：販売時の境界値						
パラメータ	確認ポイント					
上限-1	購入可能であること					
上限	購入可能であること					
上限+1	購入不可であること					

テスト設計



また、各テストのパラメーターは
変更点への影響の強さやテスト目的に応じて、『厚く確認する／代表条件を抜粋する』のメリハリをつける
→等しく網羅はしない



テスト設計

例

直接変更箇所＋ユーザーの使用にも直接関連してくる
→なるべく必要なパターンを漏れなく確認する

まずは動作していてほしい箇所/間接影響箇所
→パターン網羅ではなく、一番通りそうなところを早く確認する

将来変更しそうな箇所(リグレッションを確認したい箇所)
→パターン割り出しより、可変になるパラメータを考える



テスト設計

テスト設計例: 残数処理

→ 直接変更箇所なので、なるべくパターン漏れを少なくする
(必要に応じてテスト技法を用いてパターンを割り出す)

※残数の更新経路別で同値分割する

クラウド2が管理している残数	5-3. 時間枠別残数 (全体)
	1-3. 時間別残数情報 (当日分全体)
入場管理が管理している残数	1-2. 時間枠別残数 (現在・次・次々枠分)



パラメータ2: 設定する上限値

パラメータ	確認理由
60	今回変更される値
999	パラメータで可変の最大値
0	パラメータで可変の最小値

※要仕様確認: 本当に0が設定できて問題ないか (仕様確認後作成とする)



パラメータ3: 販売時の境界値

パラメータ	確認ポイント
上限-1	購入可能であること
上限	購入可能であること
上限+1	購入不可であること



テストケース見取り図

No	設定上限人数	購入方法	販売経路・対象枠	残数上限値	確認する境界値
1	今回変更値	発券機	現在・次・次々枠	60	上限-1
2	今回変更値	発券機	現在・次・次々枠	60	上限
3	今回変更値	発券機	現在・次・次々枠	60	上限+1
4	今回変更値	発券機	現在・次・次々枠より先の時間指定枠	60	上限-1
5	今回変更値	発券機	現在・次・次々枠より先の時間指定枠	60	上限
6	今回変更値	発券機	現在・次・次々枠より先の時間指定枠	60	上限+1
7	今回変更値	Web	Webで購入可能な時間指定枠	60	上限-1
8	今回変更値	Web	Webで購入可能な時間指定枠	60	上限
9	今回変更値	Web	Webで購入可能な時間指定枠	60	上限+1
10	最大値	発券機	現在・次・次々枠	999	上限-1
11	最大値	発券機	現在・次・次々枠	999	上限
12	最大値	発券機	現在・次・次々枠	999	上限+1
13	最大値	発券機	現在・次・次々枠より先の時間指定枠	999	上限-1
14	最大値	発券機	現在・次・次々枠より先の時間指定枠	999	上限
15	最大値	発券機	現在・次・次々枠より先の時間指定枠	999	上限+1
16	最大値	Web	Webで購入可能な時間指定枠	999	上限-1
17	最大値	Web	Webで購入可能な時間指定枠	999	上限
18	最大値	Web	Webで購入可能な時間指定枠	999	上限+1
19	最小値				

仕様確認後作成

テスト実装





テスト実装

テスト実装例: 残数処理

→ 可変箇所を固定値ではなく変数化して変更しやすくする
(テストケース上の表の値を変数化)

		境界値確認のための事前購入数			※仕様確認中のため未作成			
		上限-1を確認するための事前購入数	上限を確認するための事前購入数	上限+1を確認するための事前購入数				
可変値設定: 今回設定値	60	58	59	60				
可変値設定: 最大値	999	997	998	999				
可変値設定: 最小値	0							
設定上限人数	購入方法	販売経路・対象枠	残数上限値	確認する境界値	事前条件	操作・手順	期待結果	備考
今回変更値	発券機	現在枠 (現在・次・次々枠の代表)	60	上限-1	・対象時間枠の残数上限値が設定されている ・累計販売数が「境界値確認のための事前購入数」に示す値となっていること ・同時に他の購入処理が行われていない	1. 発券機で今すぐ入場券購入を選択する 2. 1枚購入する(決済を完了する)	購入が成立し、累計販売数が上限-1となる	
今回変更値	発券機	現在枠 (現在・次・次々枠の代表)	60	上限	・対象時間枠の残数上限値が設定されている ・累計販売数が「境界値確認のための事前購入数」に示す値となっていること ・同時に他の購入処理が行われていない	1. 発券機で今すぐ入場券購入を選択する 2. 1枚購入する(決済を完了する)	購入が成立し、累計販売数が上限となる	
今回変更値	発券機	現在枠 (現在・次・次々枠の代表)	60	上限+1	・対象時間枠の残数上限値が設定されている ・累計販売数が「境界値確認のための事前購入数」に示す値となっていること ・同時に他の購入処理が行われていない	1. 発券機で今すぐ入場券購入を選択する 2. 1枚購入する(決済を完了する)	購入が成立せず、累計販売数が上限を超えない	
今回変更値	発券機	現在・次・次々枠より先の時間指定枠	60	上限-1	・対象時間枠の残数上限値が設定されている ・累計販売数が「境界値確認のための事前購入数」に示す値となっていること ・同時に他の購入処理が行われていない	1. 発券機で時間指定入場予約購入を選択する 2. ゲスト購入にて、事前購入数を設定した対象時間枠を選択し、1枚購入する(決済を完了する)	購入が成立し、累計販売数が上限-1となる	
今回変更値	発券機	現在・次・次々枠より先の時間指定枠	60	上限	・対象時間枠の残数上限値が設定されている ・累計販売数が「境界値確認のための事前購入数」に示す値となっていること ・同時に他の購入処理が行われていない	1. 発券機で時間指定入場予約購入を選択する 2. ゲスト購入にて、事前購入数を設定した対象時間枠を選択し、1枚購入する(決済を完了する)	購入が成立し、累計販売数が上限となる	
今回変更値	発券機	現在・次・次々枠より先の時間指定枠	60	上限+1	・対象時間枠の残数上限値が設定されている ・累計販売数が「境界値確認のための事前購入数」に示す値となっていること ・同時に他の購入処理が行われていない	1. 発券機で時間指定入場予約購入を選択する 2. ゲスト購入にて、事前購入数を設定した対象時間枠を選択し、1枚購入する(決済を完了する)	購入が成立せず、累計販売数が上限を超えない	
今回変更値	Web	Webで購入可能な時間指定枠	60	上限-1	・対象時間枠の残数上限値が設定されている ・累計販売数が「境界値確認のための事前購入数」に示す値となっていること ・同時に他の購入処理が行われていない	1. Webチケットシステムで任意のアカウントでログインし、予約購入画面を表示する 2. 事前購入数を設定した対象時間枠を選択し、入場券1枚を指定して購入操作を行う。(券種は任意)	購入が成立し、累計販売数が上限-1となる	

まとめ



まとめ

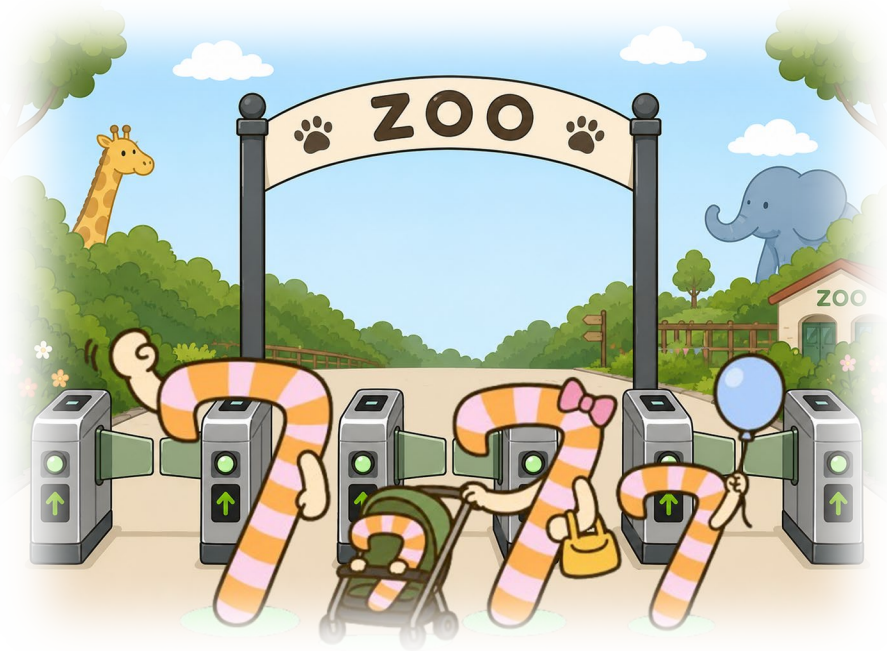


- 入場者のうれしいこと
→スムーズに入場できて、入り口で密にならないこと、人数コントロールができて、園内が密にならないこと
- 動物園事務所担当者のうれしいこと
→入場でトラブルが起きず、システムのエラー時や通信不安定時も安定して動作すること
- 動物園運営担当のうれしいこと
→スムーズな入場・購入処理により、動物園の入場やすさで魅力を損なわない
- テスト依頼側のうれしいこと
→リグレーションしやすさ

まとめ



動物園に関係するみんなのうれしさを、テストによる確認で支えることができた



ご清聴ありがとうございました。

