

# テスト設計コンテスト 決勝プレゼン資料

しち





# アジェンダ

- はじめに
- テスト設計の背景
- テスト設計で目指すこと
- 要求の実現方法1：画面全体図作成
- 要求の実現方法2：自動化に向く（メンテナンスしやすい）テスト設計
- まとめ

# はじめに



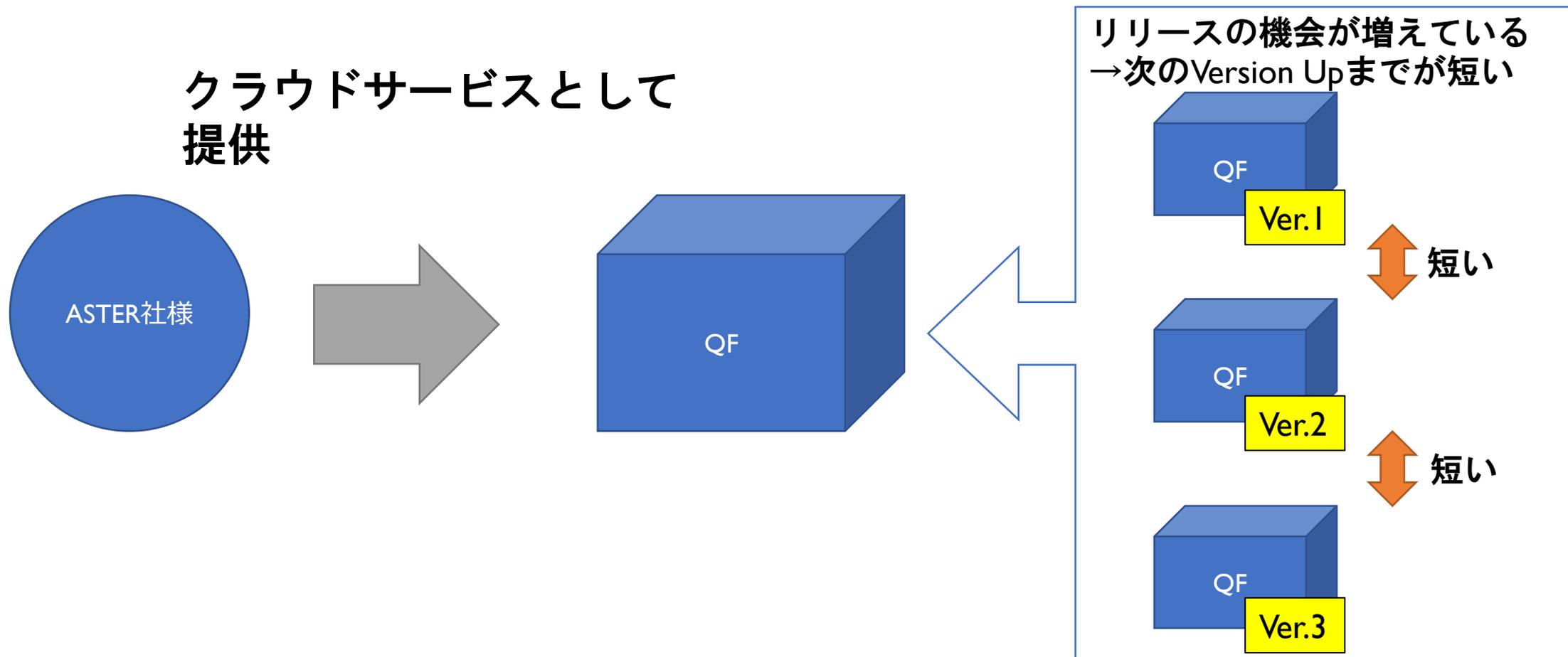
- 自己紹介  
名前：しち

職業：普通のテスター  
テスト設計コンテストは  
初めての出場



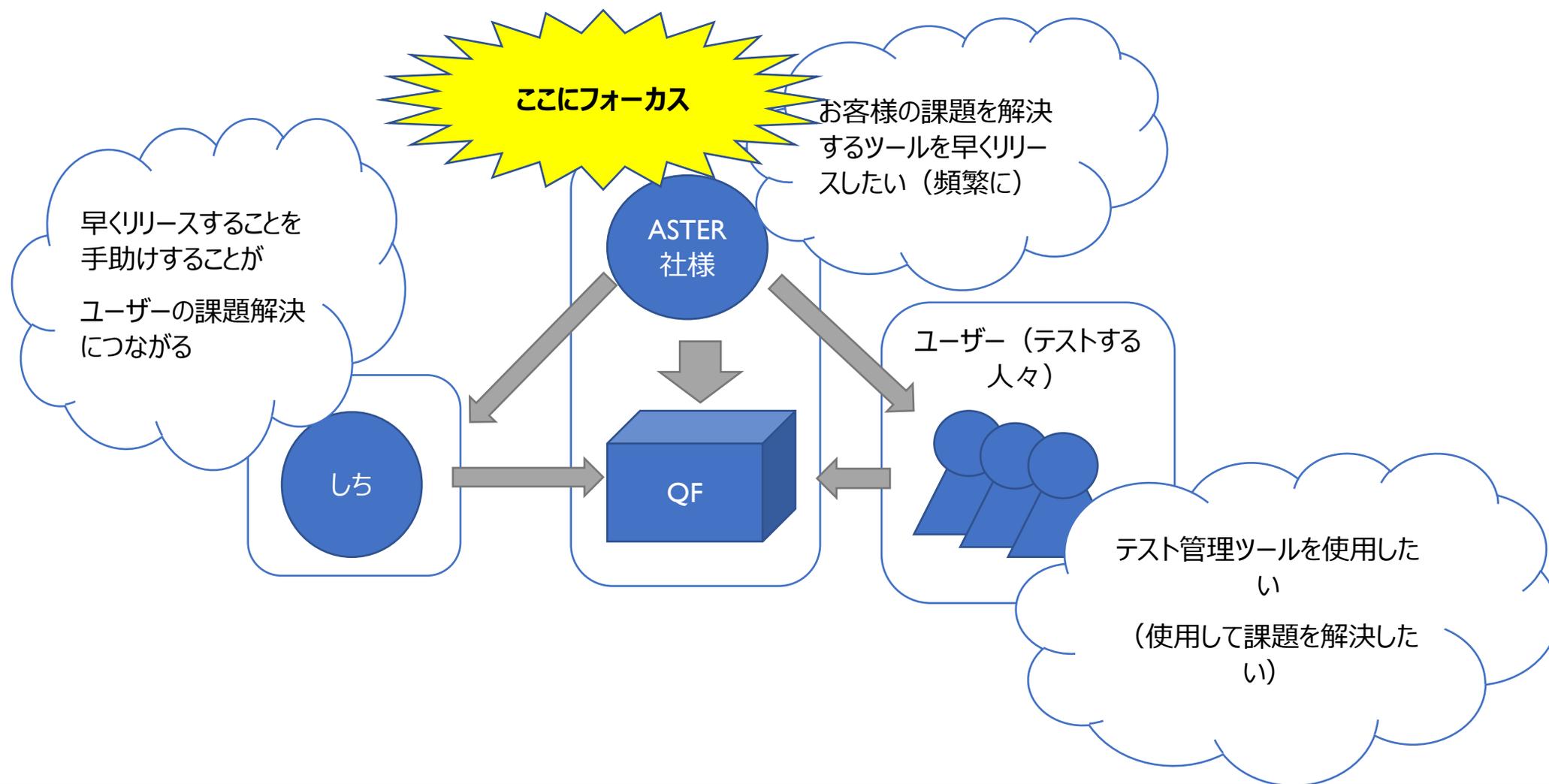


# 今回のテスト設計の背景





# ステークホルダー図





# テスト設計で目指すこと

- ・ 頻繁なリリースに耐え得るテスト設計案

- ① 確認する優先度を決めてテスト範囲絞込

- ② テスト設計プロセス効率化



# テスト設計で目指すこと

- ・ 頻繁なリリースに耐え得るテスト設計案

① 確認する優先度を決めてテスト範囲絞込

② テスト設計プロセス効率化



# テスト設計で目指すこと

毎回のリリースで  
発生するこの部分を短縮する

テスト分析・設計・実装

ソフトウェア開発全体にかかる時間

# どのように テスト設計プロセスを 効率化するか





# テスト設計で目指すこと

・テスト設計（2回目以降のリリリース）で時間がかかる箇所

①既存箇所の影響分析

②整頓されていないテストケースのメンテナンス（手動→自動化など）



# テスト設計で目指すこと

## ・ 予選から決勝の変更点

「要求→要件→テストケース」の表を削除

画面ベースの全体図を元に共通しているところを見つけ、そこからテストを組み立てる方向に一本化



# テスト設計で目指すこと

- ・ 予選から決勝の変更点

理由：

「要求→要件→テストケース」の表があったとしてテスト設計の短縮に繋がらないと考えたため



# テスト設計で目指すこと

## ・ 予選から決勝の変更点

「要求→要件→テストケース」のがあつたとしてまったく要求分析をしないわけではない

→相対的にテスト設計時間短縮が見込める割合が低い



# テスト設計で目指すこと

・テスト設計（2回目以降のリリリース）で時間がかかる箇所

①既存箇所の影響分析

→全体を俯瞰できる図を作る

②整頓されていないテストケースのメンテナンス（手動→自動化など）

→最初にある程度整頓して作る



# 画面の全体図

## ①既存箇所の影響分析

→画面全体を俯瞰できる図を作成

- ・共通箇所が分かるようにする
- ・共通している箇所は共通のテストケースを設計する



# 画面の全体図

目的：

- ・ 影響分析時間の短縮
- ・ 共通箇所重複設計防止、更新漏れ軽減



# 画面の全体図

## ■機能全体図\_詳細

この全体図を書く目的: 全体の部品を見渡し、共通している部分などを見やすくする

各画面で確認する内容を洗い出す

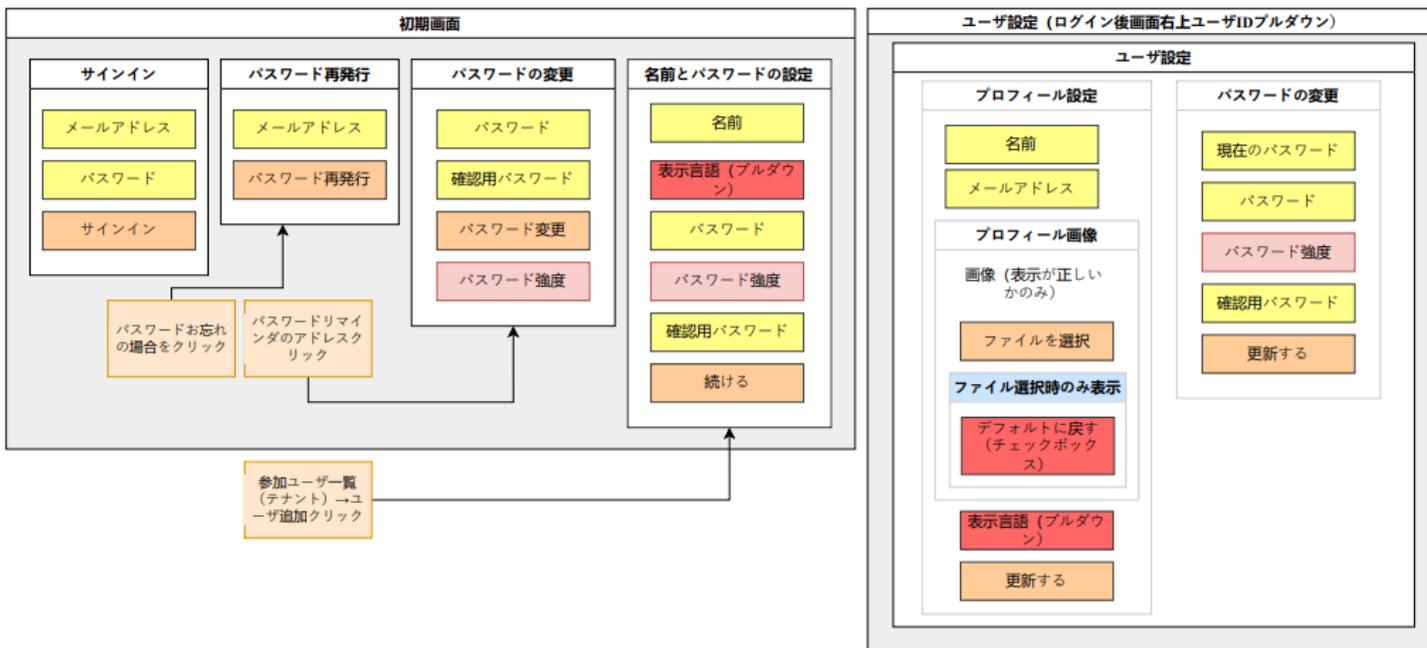
- 黄色...テキストボックス (入力可能)
- オレンジ...ボタン
- ピンク...テキストメッセージ (可変)
- グリーン...タブ
- 赤...その他の要素
- 緑点線枠...テナント管理者のみ表示
- 青点線枠...テナント管理者・プロジェクト管理者のみ表示

- 画面を漏れなく洗い出すと何が嬉しいか? 検討漏れが軽減できる
- 画面ごとで区切り、この画面でユーザーは何をしたいかを考える
- 何かを追加したいときにどこに追加するか、何が関連するかわかるようにする

### この資料の目標

後で見る人や初めて見る人が「ここは共通している」ということが一目で分かるようにする

→共通部分は別途ピックアップして記載



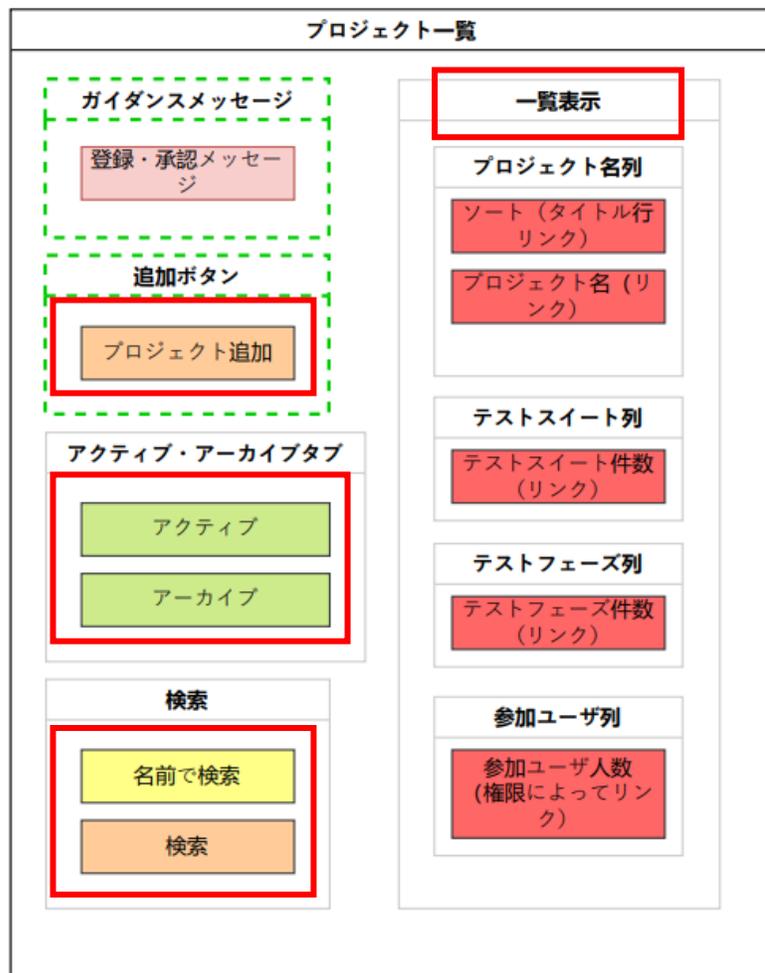
## 画面をグルーピングし

それぞれの画面の機能を貼った図を作る

共通している機能は共通した色にする。  
(一目で共通した作りを見やすくするため)



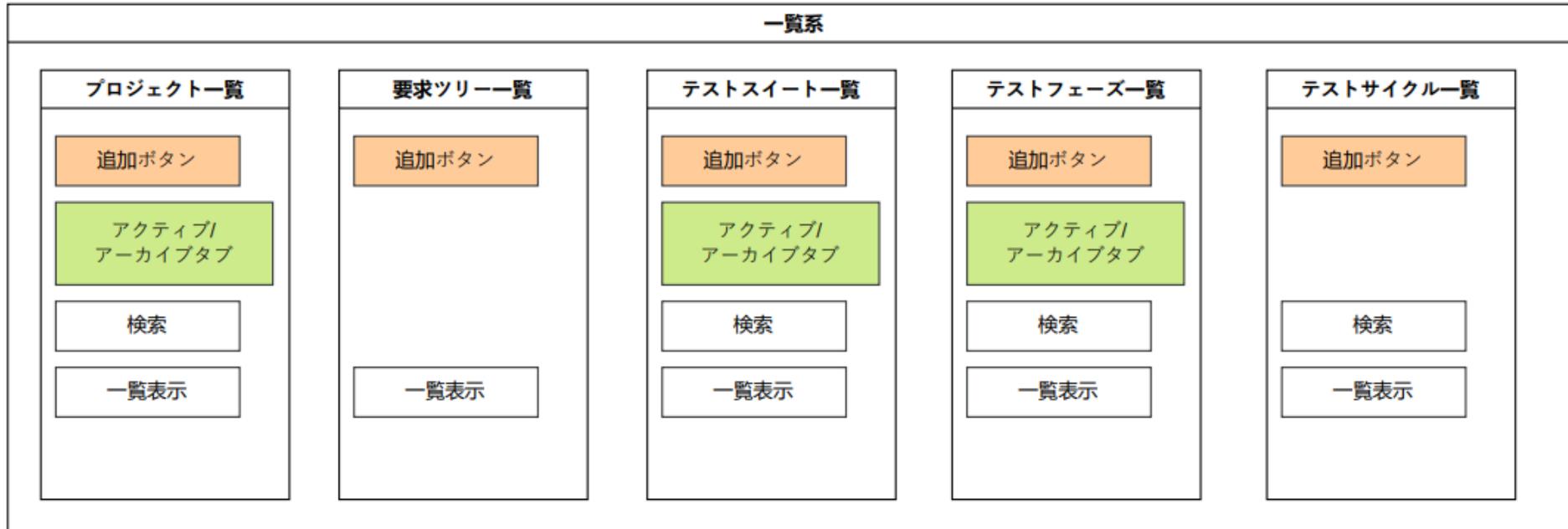
# 画面の全体図



全体図を俯瞰して、同じ構造をしている箇所を突き止める



# 画面の全体図



同じつくりをして  
いる画面をグルー  
ピングする

# 画面の全体図



		一覧系	登録・更新系	テストスイート・サイクルの紐付け系	レポート系画面	イベント一覧・新規登録系
		<p>プロジェクト一覧 追加ボタン アクティブ/アーカイブ 検索 一覧表示</p>	<p>プロジェクトの新規追加/更新 名前入力: 登録・更新 アーカイブ 削除</p>	<p>テストスイートの紐付け テストスイートの紐付け</p>	<p>レポート一覧 取束曲線グラフ グラフ範囲</p>	<p>プロジェクトイベントの新規登録 別曜日・タイトル・本文記入 登録/更新</p>
観点一覧	観点詳細	アクティブ・アーカイブ 検索 一覧表示 追加ボタン で構成される画面	名前入力 登録するで更新する 削除・アーカイブがある	テストスイートの 紐付けがある画面	取束曲線グラフ	イベントをグラフ画 面に反映する
	メールアドレスが入力されていないときにエラーすること					
	パスワードとメールの組合せがあていないときにエラーすること					
入力 (メール送信)	入力したアドレスにメールが送信されること		※一部あり			
入力 (表示反映)	入力した文字が画面に反映されること		○			○
検索	入力した文字列で検索できること	○				
	検索の反映タイミングが正しいこと (入力して即時・入力してボタン押し)	○				
	各文字の種類を検索	○	○	○	○	○
表示	固定部品の表示が正しいこと	○			○	
	可変部品の表示が正しいこと				○	
タブ	タブの切り替えが可能であること	○				
	選択したタブに従った表示になること					
機能・画面遷移	リンク・ボタンを押したときに仕様通りの画面遷移をすること	○	○	○	○	○
	入力不備があるときに画面遷移しないこと		○	○		
	チェックボックスにチェックが入れられること					
	ラジオボタンが1つずつしかチェックをつけられないこと					
	トグルボタンがトグルで表示が切り替わること					
権限ごとの表示	権限によって表示される項目が正しく反映されること	○	○			

グルーピングした画面に対して観点を分析していく

# 画面の全体図



## 観点一覧表を段階的に詳細化し、テストを実装する

テストする画面	一覧系	
観点一覧	観点詳細	テスト条件
検索	入力した文字列で検索できること	一覧に入力した文字列を含む名前がある 一覧に入力した文字列を含む名前がない
	検索の反映タイミングが正しいこと (入力して即時・入力してボタン押し)	任意の文字列1個
	各文字の種類を検索	自動化を容易にする実装→入力テストで実施
表示	固定部品の表示が正しいこと	タブ名 追加ボタン 検索のプレースホルダー 検索ボタン 一覧部分のタイトル
	可変部品の表示が正しいこと	アクティブ・アーカイブタブの一覧表示個数 一覧のプロジェクト名、件数、個数

テストする画面	一覧系	共通前提条件		
		検索結果確認に必要な内容を登録しておく ・権限の異なるユーザを登録しておく ・4件プロジェクト登録する(それぞれ含まれるテストスイート・テストフェーズの件数を覚えておく) ・1件のプロジェクトをアーカイブする ・検索欄には何も入力しない プロジェクト画面を表示しておく		
観点一覧	観点詳細	テスト条件	操作	期待動作
検索	入力した文字列で検索できること	一覧に入力した文字列を含む名前がある 一覧に入力した文字列を含む名前がない	プロジェクト名のいずれかに一致する文字列を検索欄に入力、検索を押す	入力した文字列に一致するプロジェクト名が表示される 一致しないプロジェクト名は表示されない
	検索の反映タイミングが正しいこと (入力して即時・入力してボタン押し)	任意の文字列1個	あらかじめ一覧に表示されている項目を確認しておく プロジェクト名のいずれかに一致する文字列を検索欄に入力、そのままの状態を確認	登録したプロジェクト名が全て表示されている 入力されただけではフィルタは掛からない
表示	各文字の種類を検索 固定部品の表示が正しいこと	自動化を容易にする実装→入力テストで実施	—	—
		タブ名 追加ボタン 検索のプレースホルダー 検索ボタン 一覧部分のタイトル	タブ名を確認する プロジェクトタイトルの右側を確認する 検索欄のテキストボックスのプレースホルダを確認する 検索欄のテキストボックス横を確認する プロジェクト一覧のタイトル名を確認する	左のタブがアクティブ・右のタブがアーカイブ 「プロジェクト追加」ボタンが表示されている 「名前で検索」と記載されている 「検索」ボタンが表示されている
	可変部品の表示が正しいこと	アクティブ・アーカイブタブの一覧表示個数	アクティブタブをクリックし、タブ名の横に表示されている数字を確認する アーカイブタブをクリックし、タブ名の横に表示されている数字を確認する	左から順に 「プロジェクト名」 「テストスイート」 「テストフェーズ」 「参加ユーザ」 が表示されている 一覧に表示されているプロジェクト数と表示が一致している 一覧に表示されているプロジェクト数と表示が一致している



# 画面の全体図

- ・ 共通している画面に対してのテスト → 共通画面で使い回す
- ・ 更新があったときも共通画面に対して作ったテストに追加し、共通画面全てに展開する



# 画面の全体図

何が嬉しいか？

- ・ 共通した作りをしている箇所について更新の抜け漏れが防げる

- ・ 個別にテスト設計するよりも設計工数が削減できる



# 自動化に向く設計

② 整頓されていないテストケースのメンテナンス（手動→自動化など）

→ 自動化に向く箇所は自動化前提でテストケースを書く

- ・ 自動化に向く箇所の選別
- ・ テスト条件のキーワード化・手順との分離



# 自動化に向く設計

---

目的：

- ・メンテナンス性向上



# 自動化に向く設計

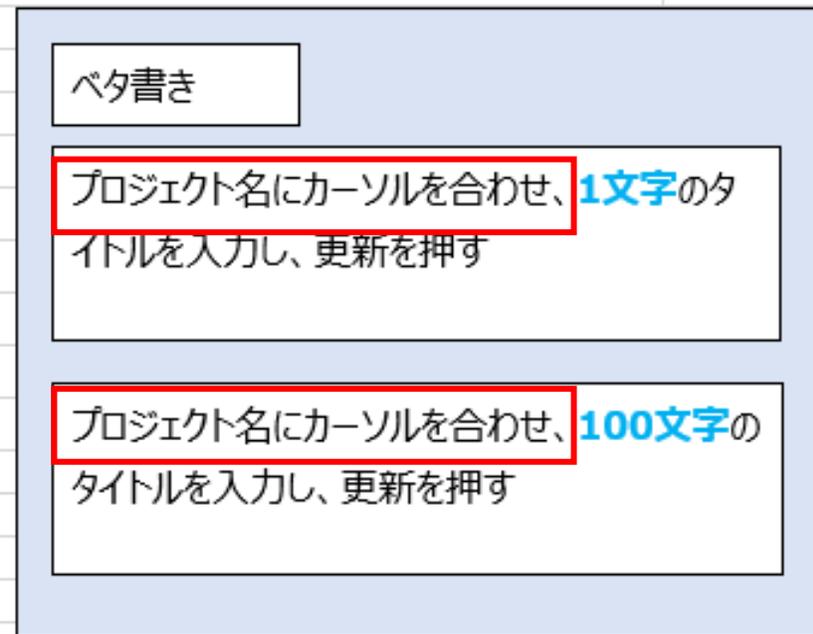
手順と条件が分離されていないテストケース

→手順が変わるたびに全てをメンテナンスしなければならず、メンテナンス性が悪い



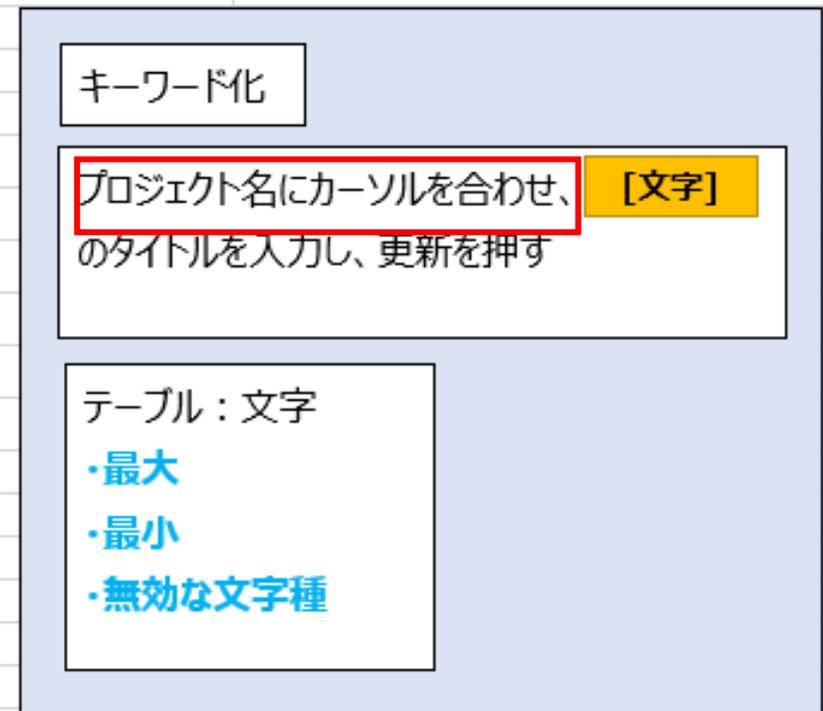
# 自動化に向く設計

手順と条件が分かれていない場合



全てのテストケースの手順を更新しなければいけない

手順と条件が分かれている場合



一か所だけの更新で済む



# 自動化に向く設計

---

自動化に向くようなテスト設計ができる  
ところ



# 自動化に向く設計

自動化に向くようなテスト設計ができる  
ところ

→ 「手順、結果の確認方法が一定」かつ  
「条件のみ複数変化する箇所」



# 自動化に向く設計

自動化に向くようなテスト設計ができる  
ところ

→ 「手順、結果の確認方法が一定」かつ  
「条件のみ複数変化する箇所」

理由：手順の変更、条件の追加どちらに  
おいてもメンテナンス箇所を少なく抑え  
ることが見込めるため



# 自動化に向く設計

以下のような箇所は除外

- 手順が複数・複雑
- 手順と条件の分離が困難
- テストの結果の観測方法が定性的



# 自動化に向く設計

## 「手順、結果の確認方法が一定」かつ 「条件のみ複数変化する」例

手順	結果
プロジェクト名に[A]を入力し更新ボタンを押す	プロジェクト名に[A]が反映されること

条件例	
条件 (A)	具体例
1文字	あ
100文字	あいうえおあいうえお… (100文字)
1000文字	あいうえおあいうえお… (1000文字)



# 自動化に向く設計

## キーワード駆動でテストできそうな箇所の分析

→ 「いずれかのデータを入力し何かをした結果を確認できるところ」

＝キーワード駆動に適した箇所とみなし、手順・条件を洗い出す



# 自動化に向く設計

## 前述の条件に当てはまる箇所の検討

大カテゴリ	小カテゴリ（確認項目）	小カテゴリ補足	確認項目深掘り
入力	入力したメールアドレスでサインイン		入力されたアドレスでサインイン アドレス形式ではない入力はエラー （再入力を促す）
	入力されたアドレスにメール送信	単一のメールアドレス	入力されたアドレスにメール送信 アドレス形式ではない入力はエラー （再入力を促す）
		複数のメールアドレス	
	入力された文字列で一覧をフィルタ		入力された文字を含むときに一覧に表示する 入力された文字を含まないときは一覧に表示しない 文字種問わずフィルタができる 検索件数
	入力された文字列を表示に反映		入力された文字列が画面に反映される 仕様外の文字が反映されない



# 自動化に向く設計

## 入力値に入れる条件の検討

条件名	この条件に含まれる目的
文字数	<ul style="list-style-type: none"><li>・仕様に定められた範囲の文字数で正しく動作すること</li><li>・仕様外入力でエラーを表示する場合はエラーメッセージが表示されること</li><li>・仕様外の文字数でユーザに著しく不利益となる動作が発生しないこと (例. 0文字・多すぎる文字で不正終了、多すぎる文字数でハングアップ)</li></ul>
文字種	<ul style="list-style-type: none"><li>・仕様に定められた範囲の文字種が正しく表示されたり動作したりすること</li><li>・仕様外入力でエラーを表示する場合はエラーメッセージが表示されること</li></ul>
文字種 (特別なコード)	<ul style="list-style-type: none"><li>・攻撃や意図しない動作に繋がる文字が無効化されていること</li></ul>
言語の種類  広義に言語ではないもの (csvやJSONなど) も含める。一定フォーマットで動作することの確認を含むため	<ul style="list-style-type: none"><li>・仕様に定められた範囲の言語の種類 (例.HTML、markdown等) で正しく動作すること</li><li>・仕様外入力でエラーを表示する場合はエラーメッセージが表示されること</li><li>・不正な書式を入力してユーザに著しく不利益となる動作が発生しないこと</li><li>・攻撃や意図しない動作に繋がる言語を入れてみて、不正動作しないこと (例.Javascript、SQL、一部のHTML)</li></ul>
フォーマット	<ul style="list-style-type: none"><li>・入力すべきフォーマットを満たしている場合は正しく動作すること</li><li>・満たしていない場合にエラーを表示する場合はエラーメッセージが表示されること</li><li>・仕様外のフォーマットでユーザに著しく不利益となる動作が発生しないこと</li></ul>



# 自動化に向く設計

## 共通する動作のグルーピング

グループ3：入力されたアドレスにメール送信(複数アドレス記載可)				
	ID	画面グループ	画面サブカテゴリ	入力部品
	I019	プロジェクト管理	プロジェクトの新規追加/設定	テストスイートのレビュアーのメールアドレス
	I042	テストフェーズ管理	テストフェーズの新規追加/設定	レビュアーのメールアドレス
グループ4：入力された文字列で一覧をフィルタ				
	ID	画面グループ	画面サブカテゴリ	入力部品
	I014	テナント	テナントのユーザー一覧	メールアドレスで検索
	I016	プロジェクト管理	プロジェクト一覧	名前で検索
	I023	プロジェクト管理	参加ユーザー一覧 (プロジェクト)	メールアドレスで検索
	I040	テストフェーズ管理	テストフェーズ一覧	名前で検索
	I056	テストサイクル管理	テストサイクル一覧	テストサイクル名で検索
	I068	ワークスペース	ワークスペース	テストサイクル名で検索
	I078	テストフェーズのチャート	進捗実績管理	テストサイクル名で検索
グループ5：入力された文字列を表示に反映				
	ID	画面グループ	画面サブカテゴリ	入力部品
	I017	プロジェクト管理	プロジェクトの新規追加/設定	プロジェクト名
	I025	要求ツリー	テスト要求ツリーの新規追加/設定	テスト要求ツリー名
	I028	テストスイート管理	テストスイートの新規追加/設定	テストスイート名
	I041	テストフェーズ管理	テストフェーズの新規追加/設定	テストフェーズ名
	I057	テストサイクル管理	テストサイクルの新規追加/設定/再テスト	テストサイクル名
	I070	テストケース編集	excelインポート	テストスイート名



# 自動化に向く設計

## 条件の詳細化

- ・ 条件の具体値検討
- ・ どの条件をどの入力テストで使用するか
- ・ 条件のカテゴリライズ（正常系・異常系）



# 自動化に向く設計

## どの条件をどの入力テストで使用するか

条件名	この条件に含まれる目的	適用するテスト技法	メール送信	フィルタ	表示反映
文字数	<ul style="list-style-type: none"><li>仕様に定められた範囲の文字数で正しく動作すること</li><li>仕様外入力でエラーを表示する場合はエラーメッセージが表示されること</li><li>仕様外の文字数でユーザーに著しく不利益となる動作が発生しないこと (例. 0文字・多すぎる文字で不正終了、多すぎる文字数でハングアップ)</li></ul>	<ul style="list-style-type: none"><li>同値分割法</li><li>境界値分析</li></ul>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
文字種	<ul style="list-style-type: none"><li>仕様に定められた範囲の文字種 (例.ASCII、全角) で正しく動作すること</li><li>仕様外入力でエラーを表示する場合はエラーメッセージが表示されること</li></ul>	<ul style="list-style-type: none"><li>同値分割法 (文字の種類をカテゴリで分類)</li></ul>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
文字種 (特別なコ)	<ul style="list-style-type: none"><li>攻撃や意図しない動作に繋がる文字が無効化されていること</li></ul>	- (テストするものの性質上グルーピングや技法で減らすことができない)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
言語の種類	<ul style="list-style-type: none"><li>仕様に定められた範囲の言語の種類 (例.HTML、markdown等) で正しく動作すること</li><li>仕様外入力でエラーを表示する場合はエラーメッセージが表示されること</li><li>不正な書式を入力してユーザーに著しく不利益となる動作が発生しないこと</li><li>攻撃や意図しない動作に繋がる言語を入れてみて、不正動作しないこと (例.Javascript、SQL、一部のHTML)</li></ul>	<ul style="list-style-type: none"><li>同値分割法 (言語の種類をカテゴリで分類)</li></ul>	攻撃や意図しない動作に繋がる言語のみ	<input type="checkbox"/>	<input type="checkbox"/>
フォーマット	<ul style="list-style-type: none"><li>入力するべきフォーマットを満たしている場合は正しく動作すること</li><li>満たしていない場合にエラーを表示する場合はエラーメッセージが表示されること</li><li>仕様外のフォーマットでユーザーに著しく不利益となる動作が発生しないこと</li></ul>	<ul style="list-style-type: none"><li>デシジョンテーブル</li></ul>	<input type="checkbox"/>	-	-



# 自動化に向く設計

## 条件の具体値検討（一例）

同値分割法による条件					
ID	条件の種類	具体値例	備考	正常系	異常系
CON_T01_01	仕様範囲外（小）	0	※仕様が定められていない場合は0文字（		○
CON_T01_02	仕様範囲内	10	※通常使用し得る値として	○	
CON_T01_03	仕様範囲外（大）	200	※仕様が定められていない場合仮置きで200文字 ここでは通常使用が考えられる値から大きい方向に外れた値を入れた時の動作を確認することが目的であるため		○
境界値分析による条件					
ID	条件の種類	具体値例	備考	正常系	異常系
CON_T02_01	仕様範囲外（小）境界（無効）	0	同値分割と重複の場合不要		○
CON_T02_02	仕様範囲外（小）境界（有効）	1	※仕様が定められていない場合も必要 文字がある・無いの境界。 1文字が0文字扱いになっているなどの不具合が潜んでいる可能性があるため	○	
CON_T02_03	仕様範囲外（大）境界（有効）	定められている場合のみ	※仕様が定められていない場合は不要 同値分割の仕様範囲外と異なり、境界値となりうる値が定められている場合の動作を確認するため	○	
CON_T02_04	仕様範囲外（大）（無効）	定められている場合のみ	※仕様が定められていない場合は不要 同値分割の仕様範囲外と異なり、境界値となりうる値が定められている場合の動作を確認するため		○



# 自動化に向く設計

## 条件の具体値検討（一例） メールアドレスのフォーマット

		1								2				3	4	5	6				7	8	9	10	11	12	13	14	15					
有効/無効		F	F	F	F	F	F	F	T	F	F	F	F	T	T	T	T	T	F	F	F	T	F	F	F	T	T	T	T	T	T	T	T	T
条件		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	
	@の前に文字列がない	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	
	@の後に文字列がない	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	
	@の前に禁止された文字がある	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y	N	N	N	N	
	@の後に禁止された文字がある	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	
	ドメインの最後がピリオドで終わる	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	
動作	「@」の前の文字列を入力してください。「入力した文字列」は完全なメールアドレスではありません								X						X	X	X	X																
	「入力した文字列」は完全なメールアドレスではありません。「@」に続く文字列を入力してください。																				X				X									
	[@の後ろに入力した文字列]内の「.」の位置が間違っています																																X	
	「@」の前の文字列に記号「任意の使用してはいけない文字」を使用しないでください。																									X	X	X	X					
	「@」に続く文字列に記号「任意の使用してはいけない文字」を使用しないでください。																													X	X			
	メール送信																																X	
	※「メール送信」についてはドメイン区切りがなくても送信できるため、この仕様で問題ないかの検討は必要と思われる																																	



# 自動化に向く設計

## 条件の具体値検討（一例）

### デシジョンテーブルで出した条件の具体値

ID	No.	条件1： 「@」前の文字列	条件2： 「@」後の文字列	条件3： 「@」前の禁止文字列	条件4： 「@」後の禁止文字列	条件5： 末尾がピリオド	具体値	動作
CON_T06_01	1	無し	無し	無し	無し	ピリオドではない	@	「@」の前の文字列を入力してください。「入力した文字列」は完全なメールアドレスではありません
CON_T06_02	2	無し	有り	無し	有り	ピリオド	@aa<a.	「@」の前の文字列を入力してください。「入力した文字列」は完全なメールアドレスではありません
CON_T06_03	3	無し	有り	無し	有り	ピリオドではない	@aa<a	「@」の前の文字列を入力してください。「入力した文字列」は完全なメールアドレスではありません
CON_T06_04	4	無し	有り	無し	無し	ピリオド	@aaa.	「@」の前の文字列を入力してください。「入力した文字列」は完全なメールアドレスではありません
CON_T06_05	5	無し	有り	無し	無し	ピリオドではない	@aaa	「@」の前の文字列を入力してください。「入力した文字列」は完全なメールアドレスではありません
CON_T06_06	6	有り	無し	有り	無し	ピリオドではない	a\aa@	「入力した文字列」は完全なメールアドレスではありません。「@」に続く文字列を入力してください。
CON_T06_07	7	有り	無し	無し	無し	ピリオドではない	aaa@	「入力した文字列」は完全なメールアドレスではありません。「@」に続く文字列を入力してください。
CON_T06_08	8	有り	有り	有り	有り	ピリオド	a\aa@aa<a.	「@」の前の文字列に記号「任意の使用してはいけない文字」を使用しないでください。
CON_T06_09	9	有り	有り	有り	有り	ピリオドではない	a\aa@aa<a	「@」の前の文字列に記号「任意の使用してはいけない文字」を使用しないでください。
CON_T06_10	10	有り	有り	有り	無し	ピリオド	a\aa@aaa.	「@」の前の文字列に記号「任意の使用してはいけない文字」を使用しないでください。
CON_T06_11	11	有り	有り	有り	無し	ピリオドではない	a\aa@aaa	「@」の前の文字列に記号「任意の使用してはいけない文字」を使用しないでください。
CON_T06_12	12	有り	有り	無し	有り	ピリオド	aaa@aa<a.	「@」に続く文字列に記号「任意の使用してはいけない文字」を使用しないでください。
CON_T06_13	13	有り	有り	無し	有り	ピリオドではない	aaa@aa<a	「@」に続く文字列に記号「任意の使用してはいけない文字」を使用しないでください。
CON_T06_14	14	有り	有り	無し	無し	ピリオド	aaa@aaa.	[@の後ろに入力した文字列]内の「.」の位置が間違っています
CON_T06_15	15	有り	有り	無し	無し	ピリオドではない	aaa@aaa	メール送信



# 自動化に向く設計

## テスト実装例

■テスト実装				
テストスイートID	テスト概要	目的		
INPT_TS001	メールアドレス入力-サインイン	メールアドレスでサインインできることを確認する 仕様外の文字列を入力したときにユーザに致命的となる動作が起きないことを確認する フォーマットに沿っていない文字列を入力したときに適切なエラーメッセージが表示されることを確認する		
参照テーブル	前提条件	操作1	操作2	結果
I001	IDのテーブルの「画面グループ」-「画面サブカテゴリ」の画面を表示する パスワード欄に「仕様範囲内」でサインインできるパスワードを入力しておく	「入力部品」にカーソルを合わせる	テーブル「条件」の「具体値」を入力し、「サインイン」を押す	テーブル「条件」各行参照
テーブル：画面				
ID	画面グループ	画面サブカテゴリ	入力部品	
I001	初期画面	サインイン	メールアドレス	
テーブル：条件				
ID	条件の種類	具体値	結果	
CON_T01_01	仕様範囲外（小）	（何も入力しない）	サインインできない	
CON_T01_02	仕様範囲内	10文字程度のサンプルメールアドレス	サインインできる	
CON_T01_03	仕様範囲外（大）	200文字程度のサンプルメールアドレス	サインインできる	



# 自動化に向く設計

何が嬉しいか？

- ・更新する箇所の煩雑化が防げる（手順なら一か所、条件ならテーブルに追加するだけでよくなる）
- ・将来テストを自動化するときにキーワード駆動のテストを実装しやすくなる



# まとめ

- ・ 全体図の作成で影響範囲の分析をしやすくする
- ・ 共通している部分はテスト設計を共通にする
- ・ 条件を明確に分けられそうなところは手順と条件を分けてテストケースを作成しておく

これらによってテスト設計をスムーズにし、頻繁なリリースに耐えられるようにすることができる。

ご清聴ありがとうございました。

