ASTER通信カラオケシステム
テスト設計のご提案

2017年2月24日(土)

いわた、すはら、なかはら、
えのき、おーだん

私たちのチーム紹介

こすにゃん V3

おーだん

なかはら

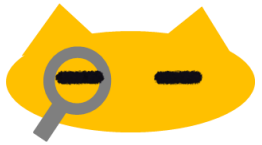
いわた

すはら

えのき

東海/関西のテスト設計の有志。

アジェンダ

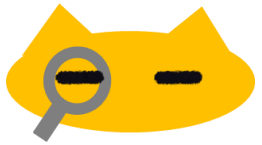


1. 提案の概要
2. 前提条件と課題
3. テストの全体像
4. まとめ

テスト開発コンセプト

「製品のリリースを止めるバグ
を許容範囲まで減らす！」

前提条件



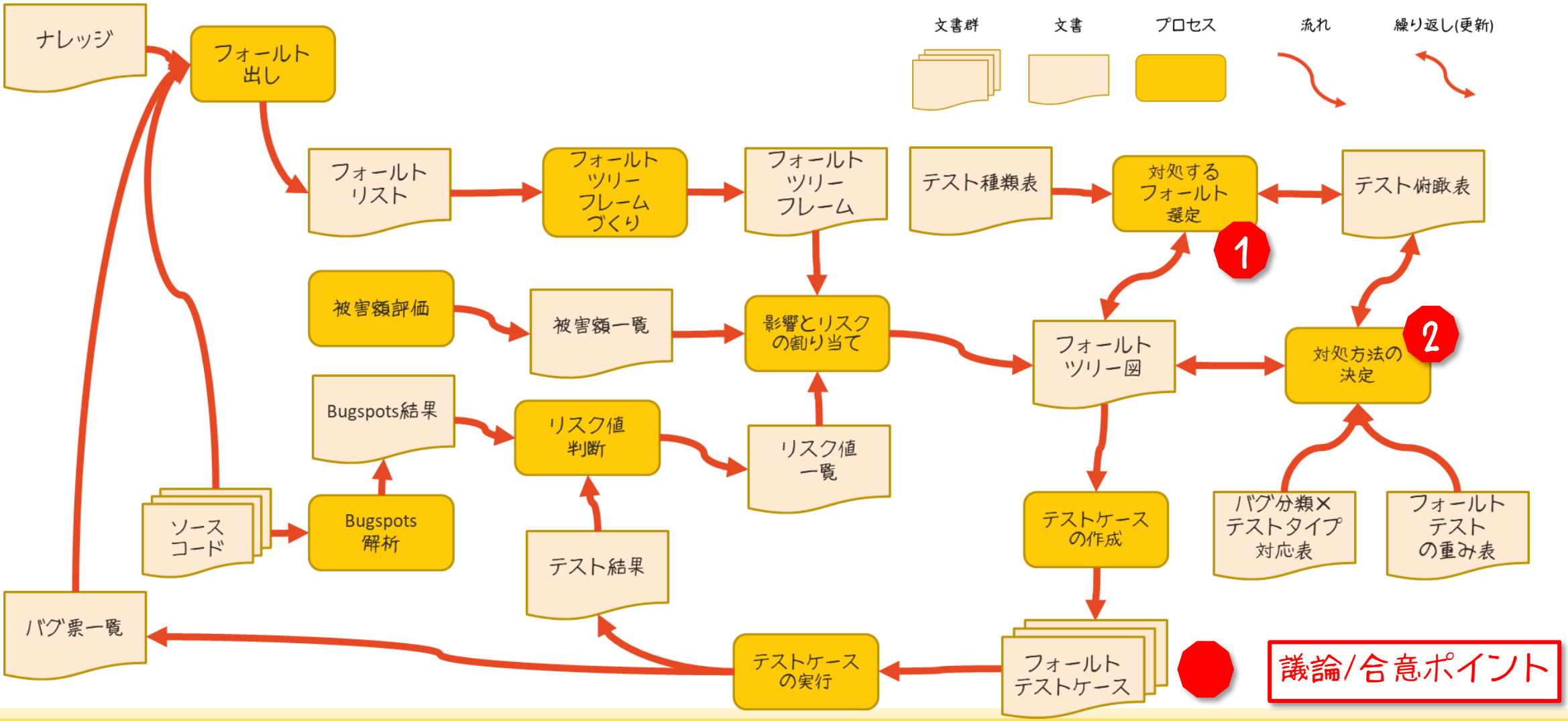
- 開発製品　　:オープンソースソフトウェアの通信カラオケシステム
- 立場　　:コミュニティメンバ QA担当
- リソース　　:2人
- リリース周期　:2週間(最新版)、6か月(安定板)

製品のフィーチャーはコミュニティ開発者の意思で、適宜実装される。

リリースについての課題

1. 製品のリリースを止める事象およびバグをどう特定するか？
2. テストで対処する/しない事象はどのように決めるか？
3. 対処するためにどんなテストをどれくらい行うか？

テストの全体像


ナレッジ
フォールト出し
フォールトリスト
フォールトツリーフレームづくり
フォールトツリーフレーム
テスト種類表
対処するフォールト選定
テスト俯瞰表
被害額評価
被害額一覧
影響とリスクの割り当て
フォールトツリー図
対処方法の決定
Bugspots結果
リスク値判断
リスク値一覧
ソースコード
Bugspots解析
テスト結果
テストケースの作成
バグ分類×テストタイプ対応表
フォールトテストの重み表
バグ票一覧
テストケースの実行
フォールトテストケース
文書群
文書
プロセス
流れ
繰り返し(更新)
1
2
議論/合意ポイント

テストアーキテクチャ(フォールト)

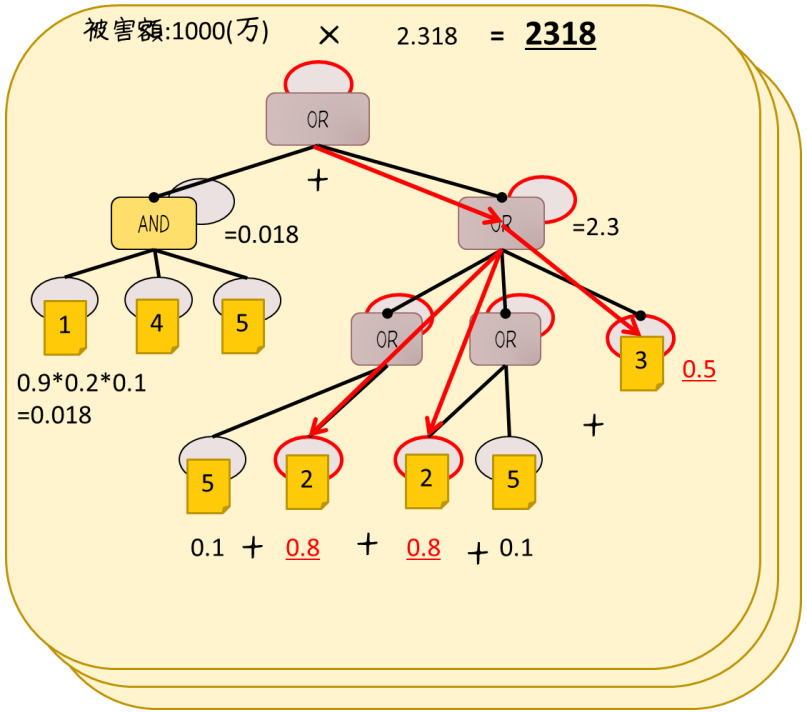
1

事象の影響度(被害金額)と発生確率(リスク値)から

対処すべき事象(フォールト)を特定し、対応者について合意する。

身体的リスク	1000万円
金銭的リスク	500万
社会的要請	5億円
物理的破壊リスク	10万円

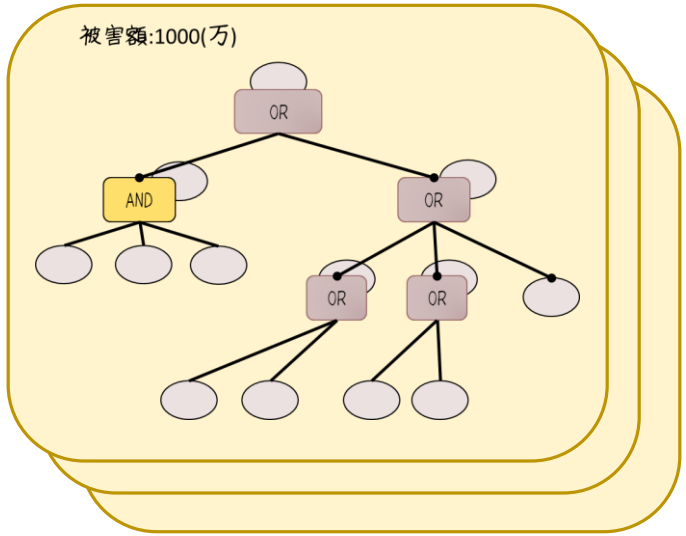
身体に影響	不快な音波
	大音量
	気分が悪くなる映像が流れる
	チカチカ(光の点滅)
課金	課金ができない
	課金の計算が正しくない
	課金データが改ざんされる
社会的要請	クラッキングされる
	踏み台にされる
	操作データが外部にもれる
	楽曲データがとられる
物を破壊	ハードウェアが壊れる
	バックアップデータが壊れる
	サーバーのデータを破壊する
	大量にデータをDL



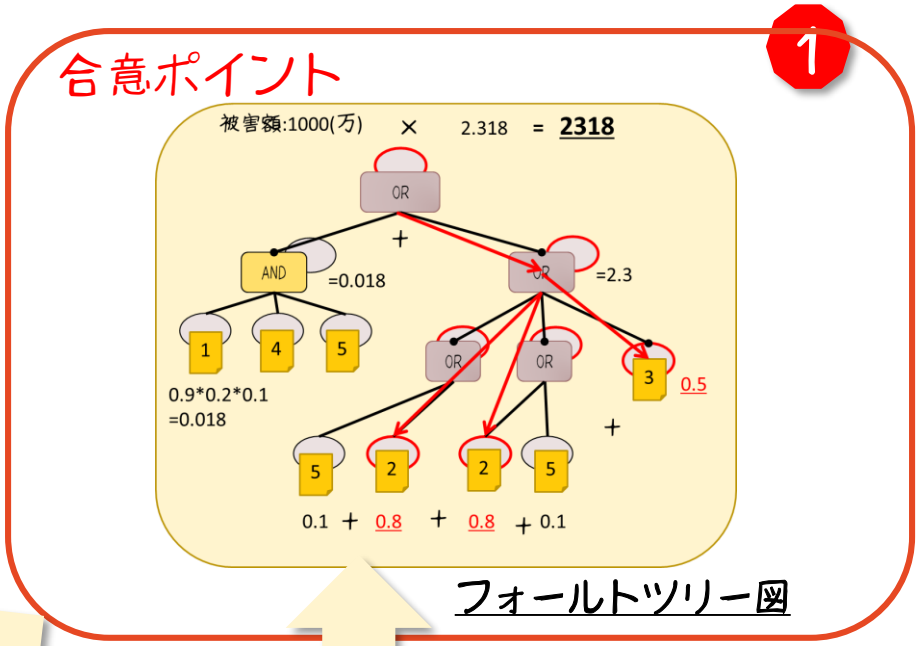
テストアーキテクチャ(フォールト)

身体的リスク	不快な音波
	大音量
	気分が悪くなる映像が流れる
	チカチカ(光の点滅)
金銭的リスク	課金ができない
	課金の計算が正しくない
	課金データが改ざんされる
社会的要請	クラッキングされる
	踏み台にされる
	操作データが外部にもれる
	楽曲データがとられる
物理的破壊リスク	ハードウェアが壊れる
	バックアップデータが壊れる
	サーバーのデータを破壊する
	大量にデータをDL

フォールトリスト



フォールトツリーフレーム



フォールトツリー図

合意事項:

1. 事象の被害金額とリスク値
2. 対処する起こってほしくない事象
3. 起こってほしくない事象の対応者

身体的リスク	1000万円
金銭的リスク	500万
社会的要請	5億円
物理的破壊リスク	10万円

被害額一覧

No.	File Name	Path	Bugspots (0.0~1.00)	前回の値	更新の有無
1	pykdb.py	\pykdb.py	0.0134	None	更新
2	pykaraoke.py	\pykaraoke.py	0.0093	None	更新
3	ChangeLog	\ChangeLog	0.0078	None	更新
4	pykar.py	\pykar.py	0.0024	None	更新
5	pycdg.py	\pycdg.py	0.0019	None	更新
6	performer_prompt.py	\performer_prompt.py	0.0018	None	更新
7	pympg.py	\pympg.py	0.0010	None	更新
8	_pycdgAux.c	_pycdgAux.c	0.0009	None	更新
9	setup.py	\setup.py	0.0009	None	更新
10	pykmanager.py	\pykmanager.py	0.0006	None	更新
11	.cvsignore	\.cvsignore	0.0003	None	更新
12	pykaraoke_mini.py	\pykaraoke_mini.py	0.0002	None	更新
13	pykplayer.py	\pykplayer.py	0.0002	None	更新
14	README.txt	\README.txt	0.0002	None	更新

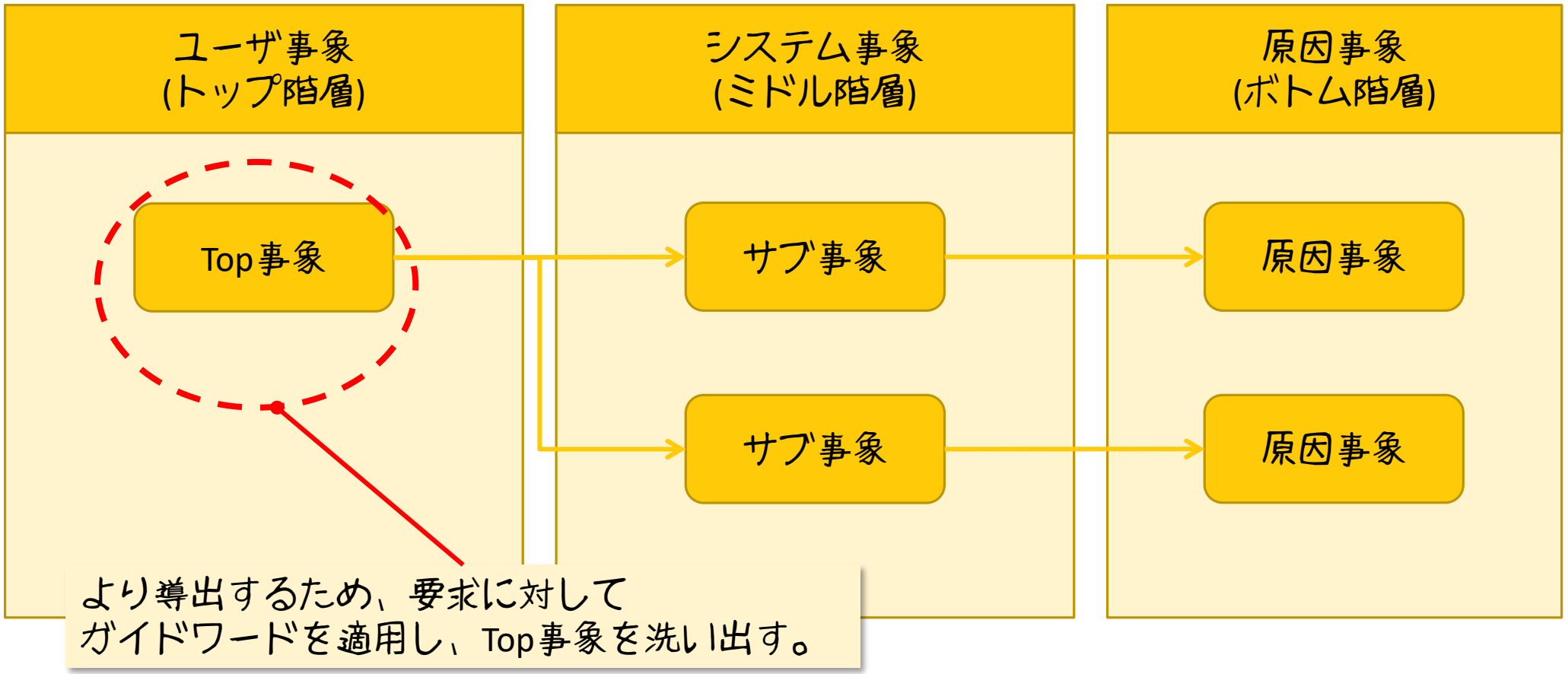
リスク値一覧

フォールトツリー作成方法

三層に分けてフォルトツリーを考える

フォールトツリー作成方法

ステークホルダーの要求からTop事象を出す。



要求からのTop事象(フォールト)だし

要求＋HAZOPガイドワードにより、Top事象を効率よく導出する。

オーナ

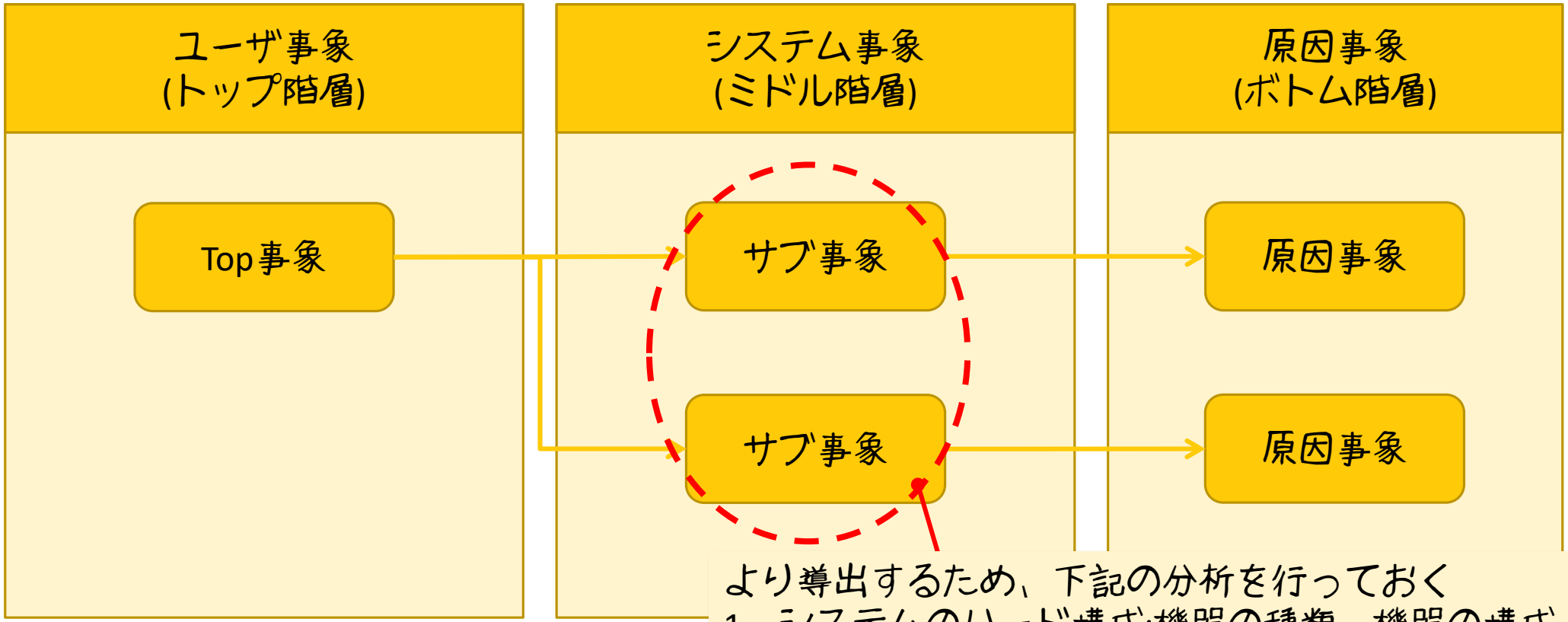
お客さんが利用してくれ、儲かる（課金ができる）

種類：違う

課金額を誤る

フォールトツリー作成方法

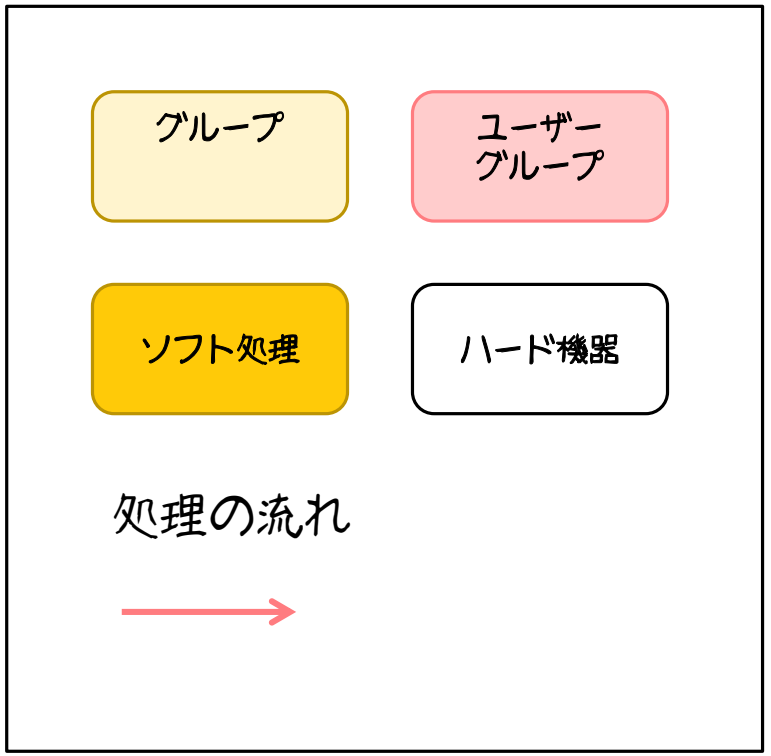
また他にハードとソフトの分析を行ってから作成する。



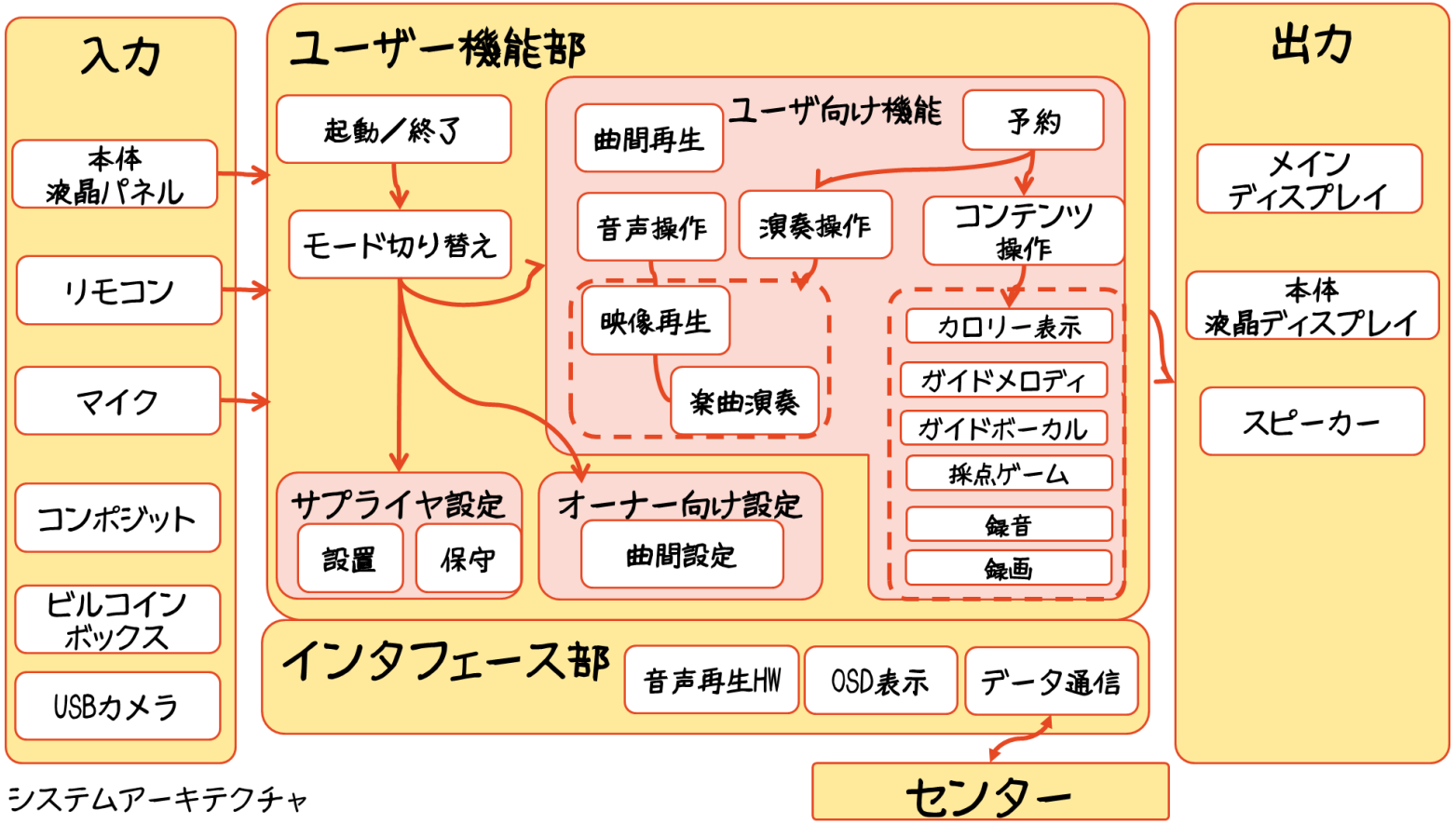
より導出するため、下記の分析を行っておく
1．システムのハード構成:機器の種類、機器の構成
2．システムのソフト構成:処理の種類、処理実装のファイル構成
※想定でもよいので出す

ハードの分析:機器の構成

ハードウェア機器の構成を考える。



例



システムアーキテクチャ

ソフトの分析:処理実装のファイル構成

ソフトウェア処理実装のファイル構成を考える。

ファイル

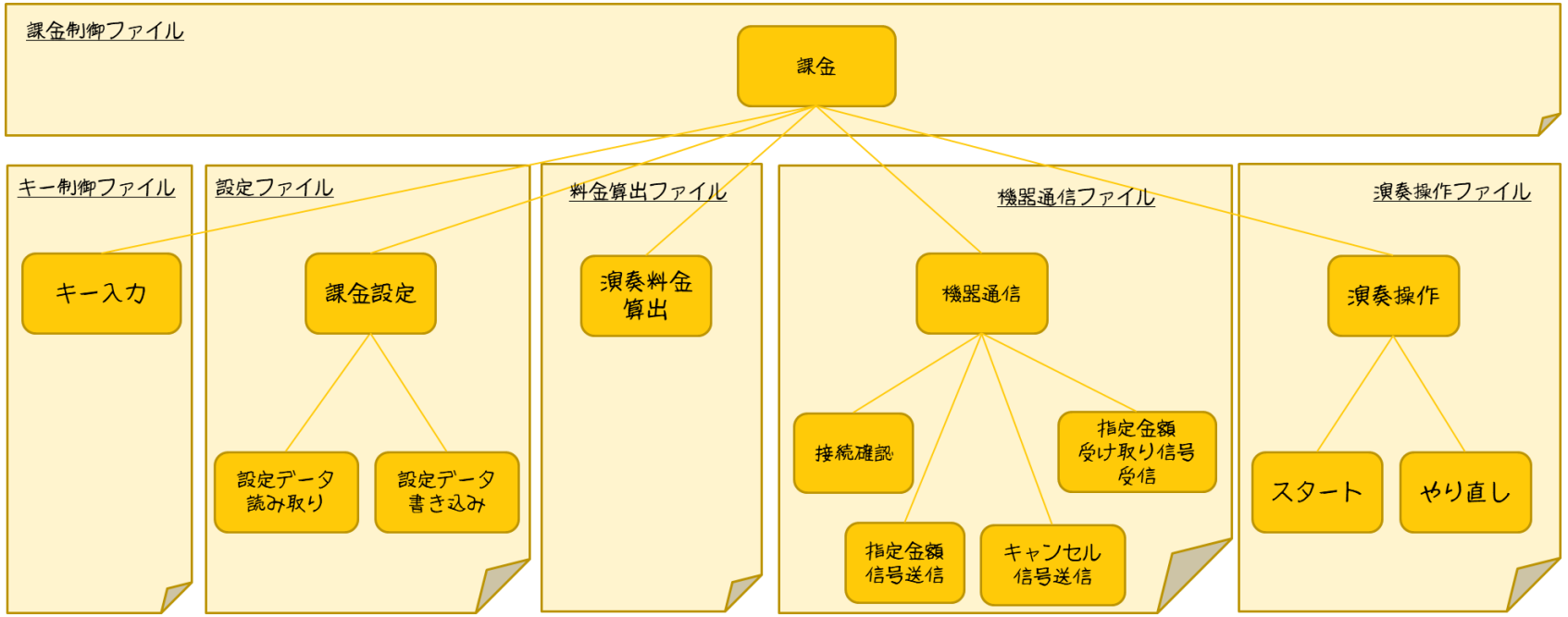
課金制御ファイル

処理

ソフト処理

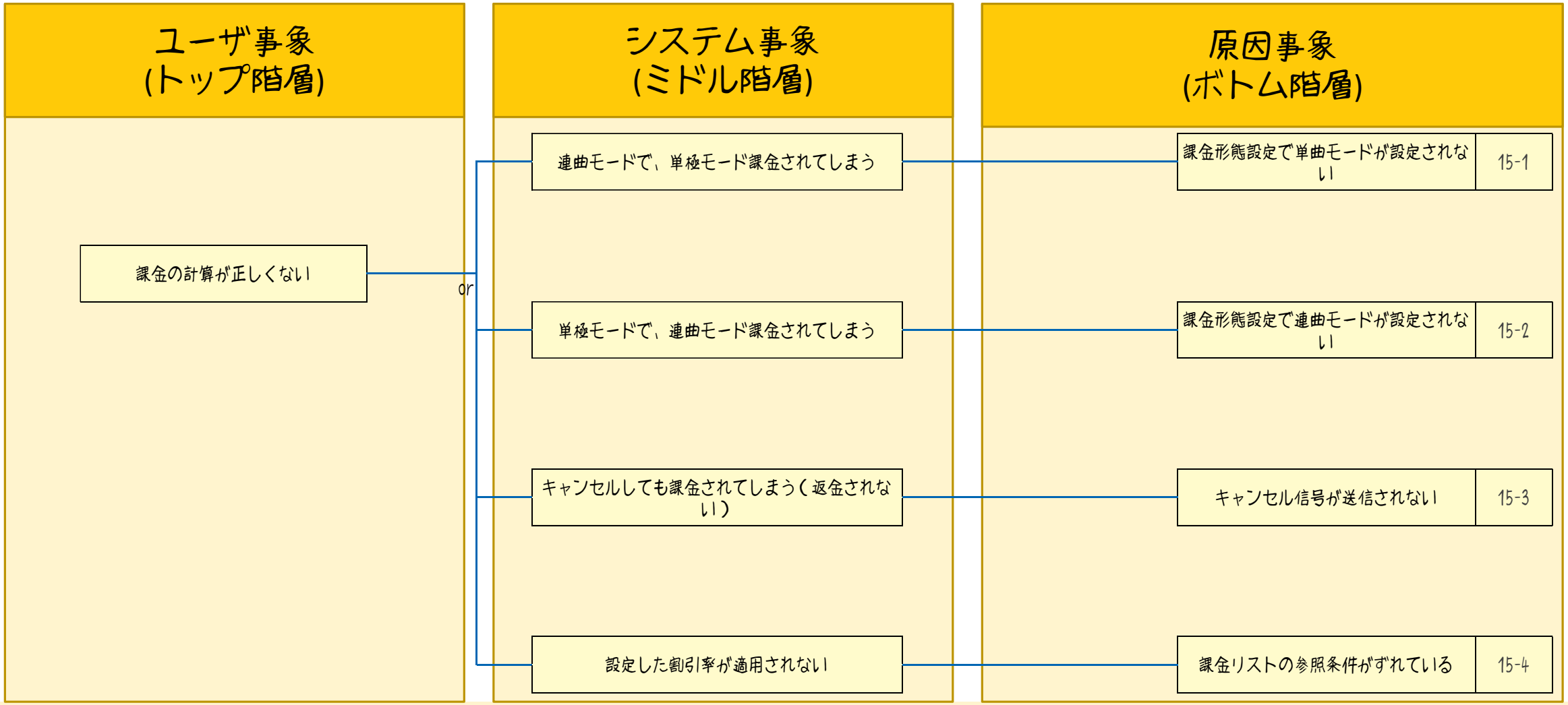
関連

例



※可能であれば、処理の流れも書く。

フォールトツリーフレームサンプル

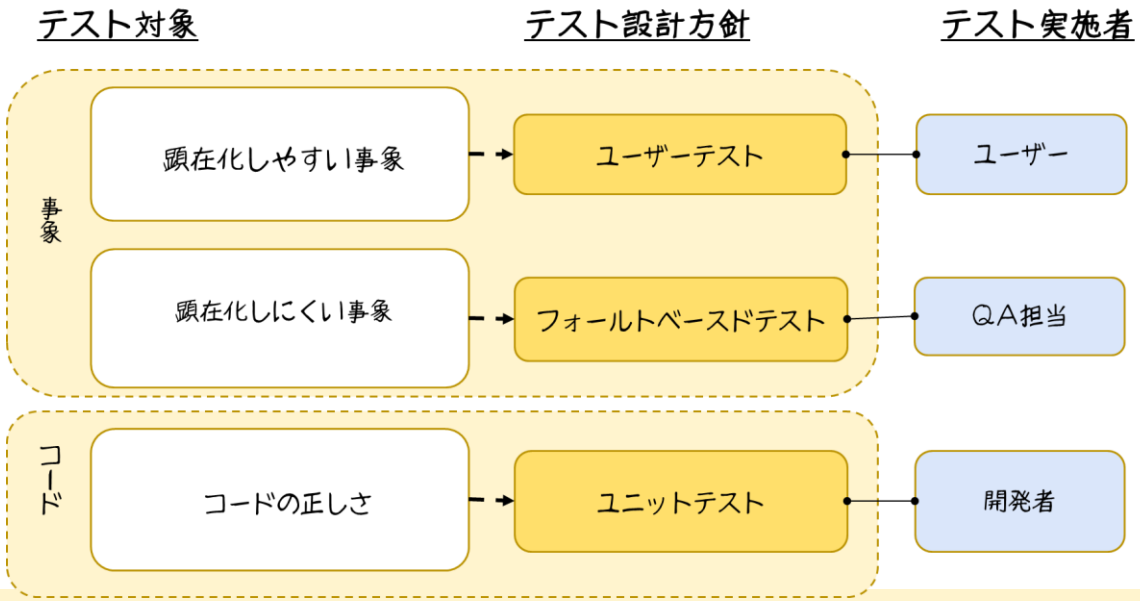


テストアーキテクチャ(テスト設計方針)

テストで対処する起こってほしくない事象のために、テスト設計方針を合意する。

合意事項:

1. テストにおける対応者
2. テストの重み(カバレッジ)
3. テストタイプ(対応方法)

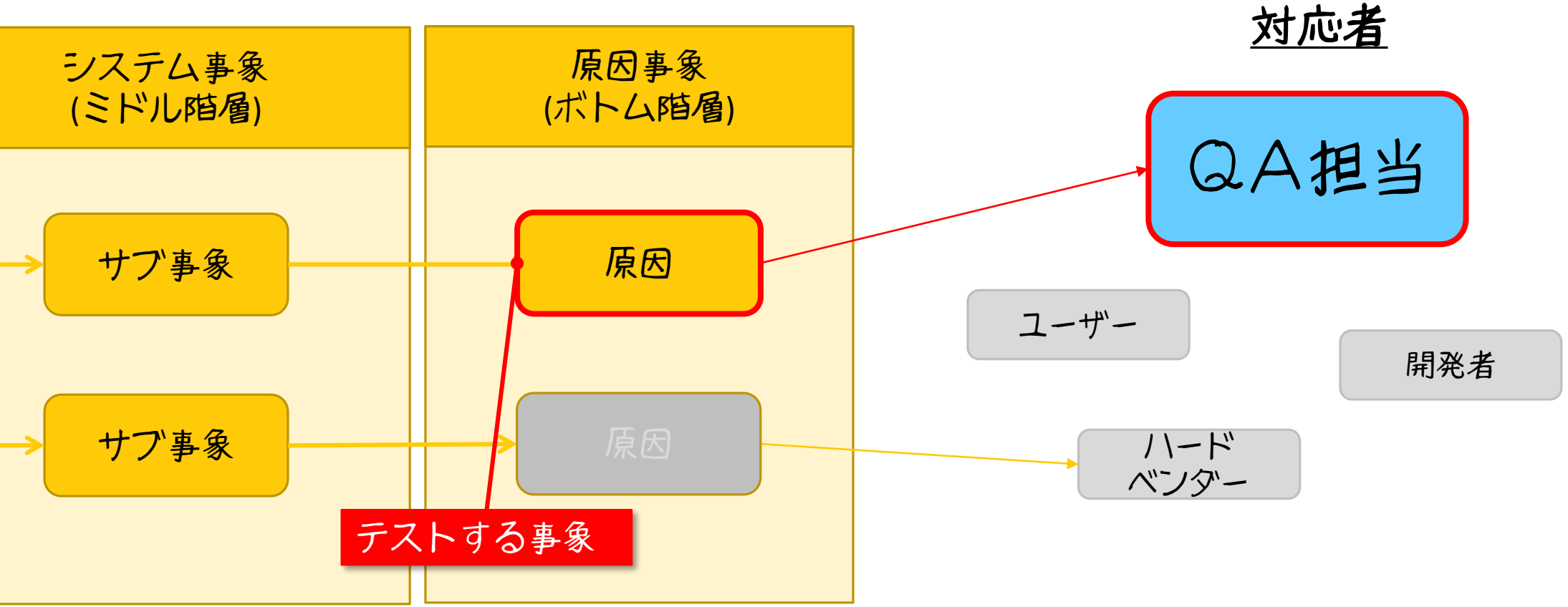


		テストタイプ※1				
		データ	条件	シナリオ	イベント/タイミング	構成
テストの重みづけ	軽 最低限のテスト	有効/無効	重複結果なし	C0	0-switch	1-wise
	中	＋同値クラス	無効なし	C1	セル網羅	2-wise
	重	＋境界値	完全	C2	1-switch	3-wise

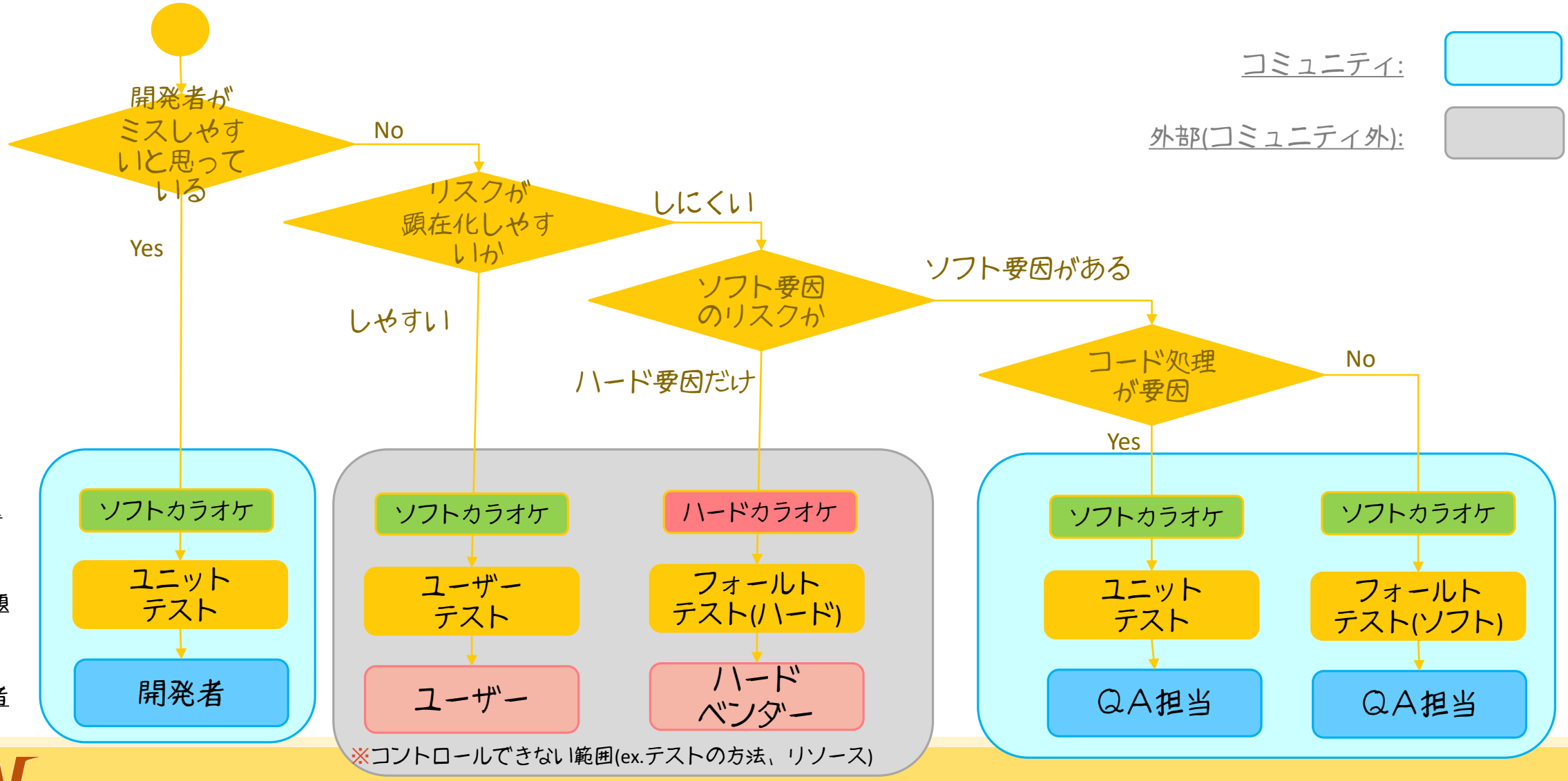
		テストタイプ				
		データ	条件	シナリオ	イベント/タイミング	構成
リスク値低減度	軽 最低限のテスト	31.00%	25.00%	56.76%	61.67%	23.57%
	中	71.00%	45.00%	77.35%	83.89%	80.71%
	重	95.00%	95.00%	95.00%	95.00%	95.00%

対処する事象(フォールト)の選定と対処方法割り当て

QAが対処すべきかを合意する。

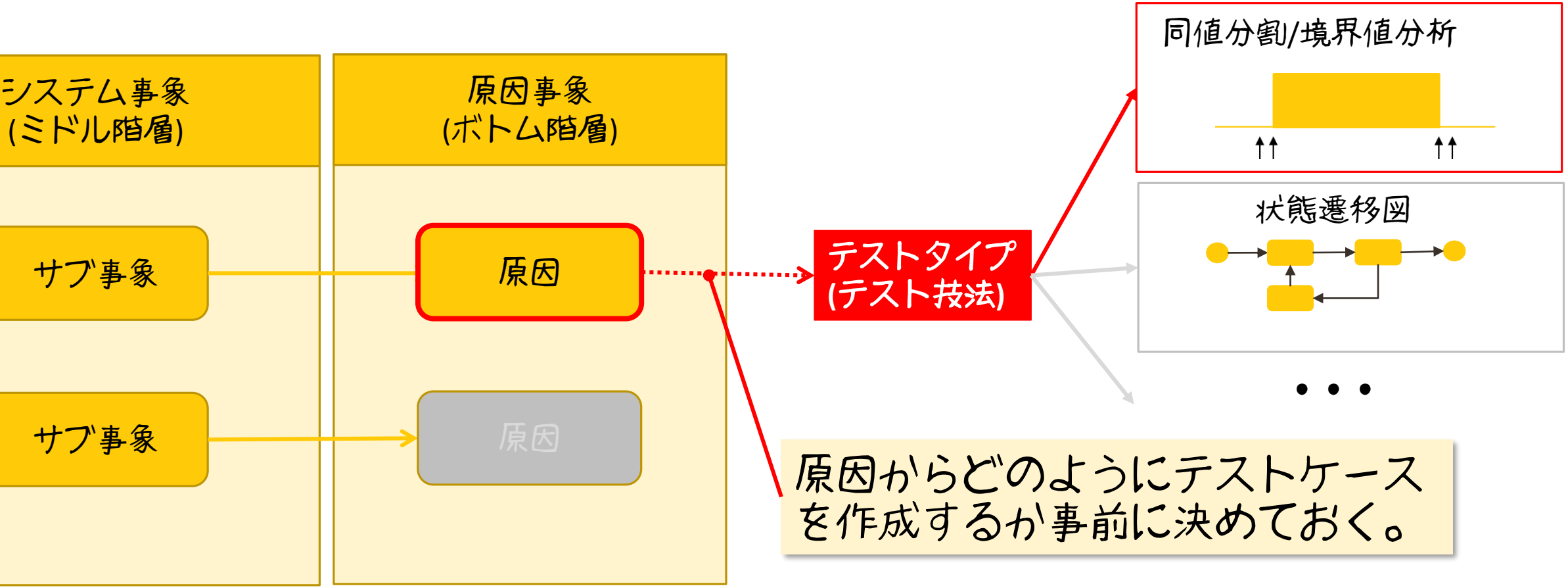


事象ごとのテスト担当決めフロー



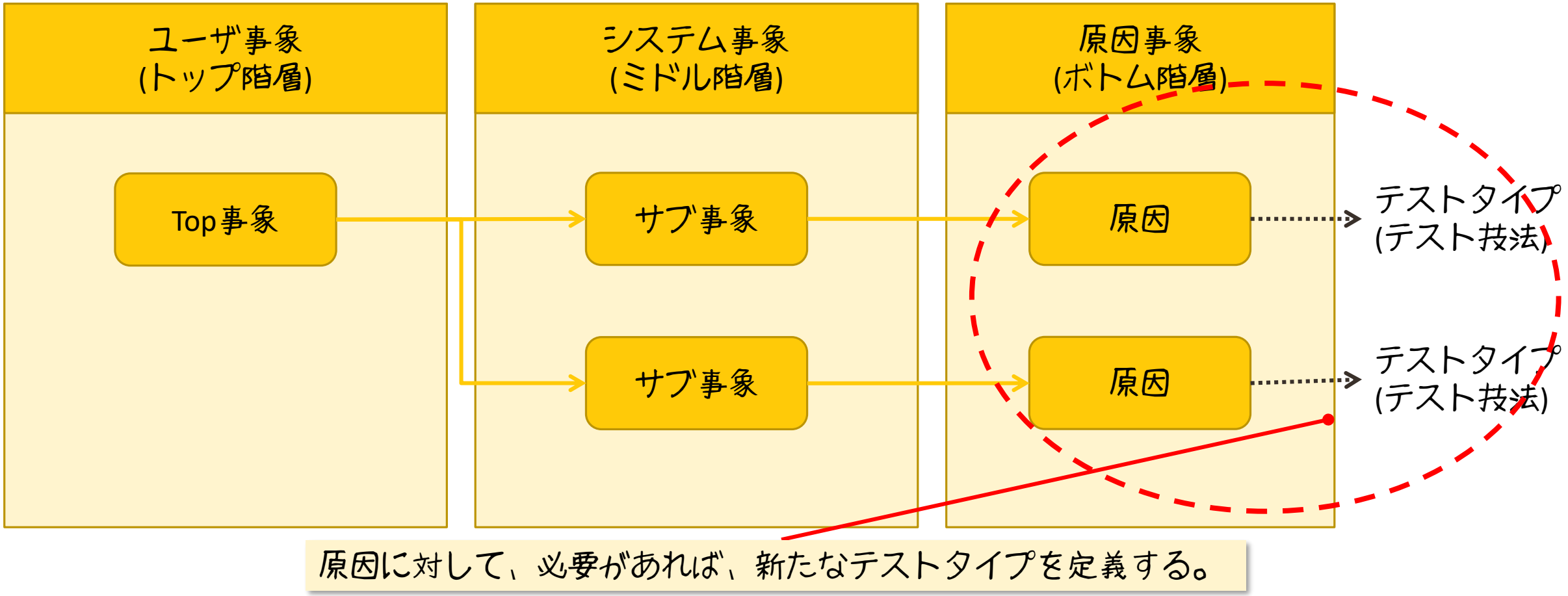
対処する事象(フォールト)の選定と対処方法割り当て

QAが対処すべき事象に対処方法（テストタイプ）を割り当てる。

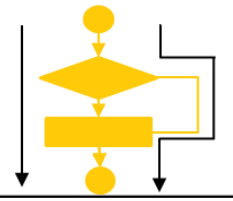
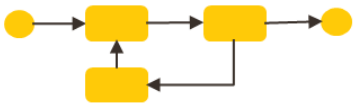
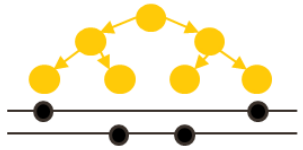


テストケース作成方法

原因からどのようにテストケースを作成するか決めておく。



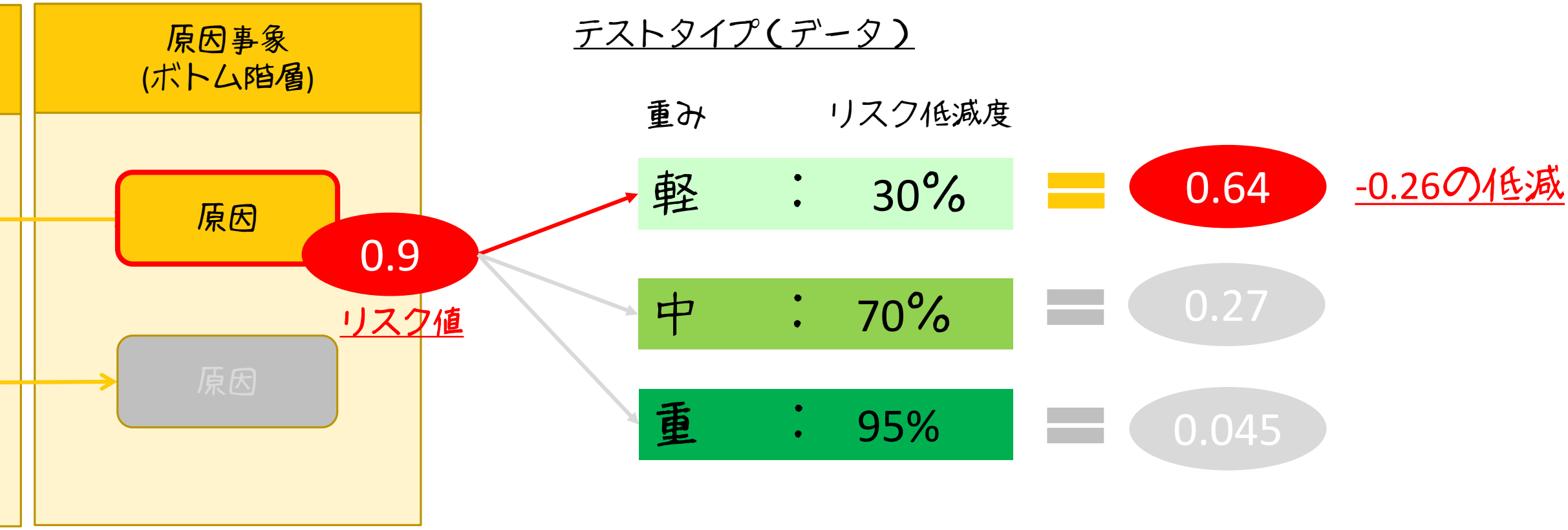
テストタイプ⇔テスト技法対応表

テストタイプ	(キーワード)	テスト技法	(テストケース導出モデル)
• データ (領域、境界、サイズ) • 性能	テストタイプ+ 抜け/漏れ ズレ/跳び 過剰/不足 矛盾/衝突 未反映/初期値/なし	同値分割 境界値分析	例 
• 流れ (シナリオ、処理)	同上	フローチャート	例 
• イベント/タイミング (状態、モード、遷移)	同上	状態遷移図/表	例 
• 構成 (環境、機器、モジュール)	同上	分類ツリー法	例 
• 条件 (入出力、制約)	同上	デシジョンテーブル	例

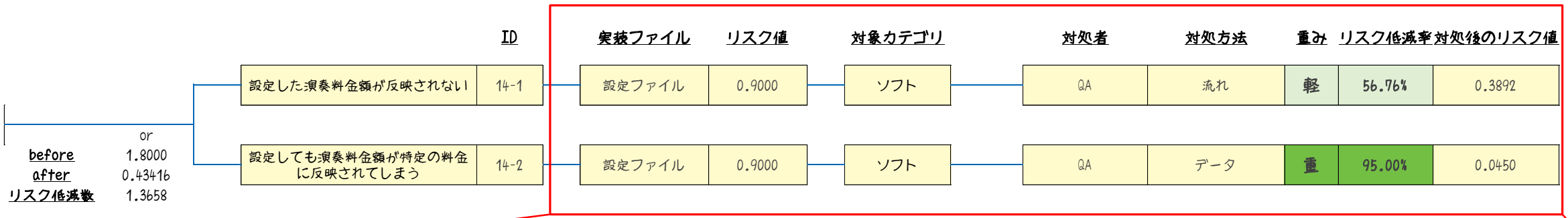
	1	2	3	4
A	F	F	T	T
B	F	T	F	T
Result		○	○	○

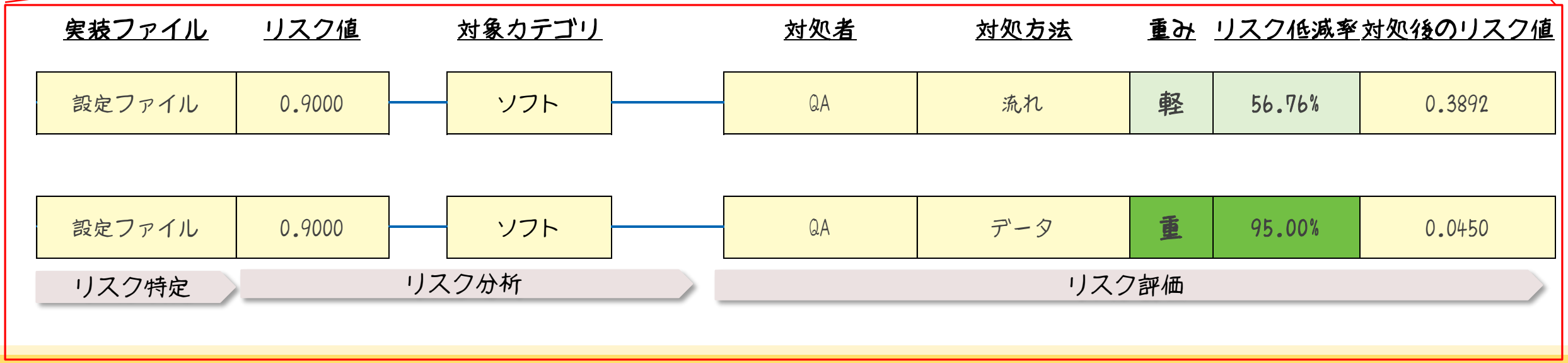
対処する事象(フォールト)の選定と対処方法割り当て

テストタイプの重みを割り当て、事象全体の「やばさ」を下げる。



フォールトツリー図サンプル



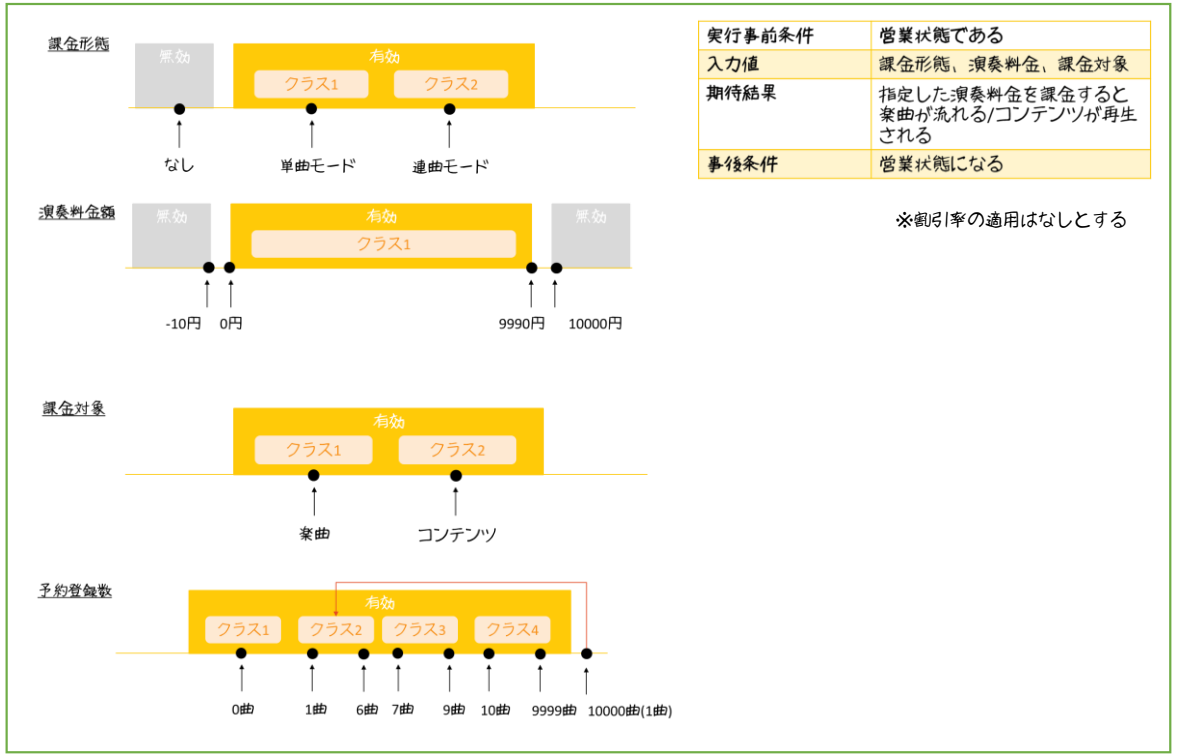


テストケースサンプル

テストケース導出モデルからテストケースを作成する。

テストケース導出モデル

テストタイプ	データ
テスト技法	同値分割/境界値分析



実行事前条件	営業状態である
入力値	課金形態、演奏料金、課金対象
期待結果	指定した演奏料金を課金すると楽曲が流れる/コンテンツが再生される
事後条件	営業状態になる

※割引率の適用はなしとする

テストケース導出モデル

テストケース

重み	重
カバレッジ	有効/無効＋同値クラス＋境界値
実施テストケース数	168

テストケースNo.	入力条件（事前状態）					期待結果	事後条件	事象ＩＤ
	課金形態	演奏料金額	課金対象	予約登録数				
1	連曲モード	-10円	コンテンツ	1曲		設定できない	営業状態になる	14-1
2	単曲モード	-10円	コンテンツ	1曲		設定できない	同上	16-1
3	連曲モード	-10円	コンテンツ	10000曲		設定できない	同上	
4	なし	10000円	楽曲	9999曲		設定できない	同上	
5	連曲モード	10000円	コンテンツ	10000曲		設定できない	同上	
6	なし	0円	コンテンツ	6曲		課金されない	同上	
7	単曲モード	10000円	楽曲	9曲		設定できない	同上	
8	単曲モード	0円	コンテンツ	7曲		指定金額で課金される	同上	
9	単曲モード	0円	楽曲	6曲		指定金額で課金される	同上	
10	連曲モード	0円	コンテンツ	7曲		指定金額で課金される	同上	
11	単曲モード	9990円	楽曲	0曲		指定金額で課金される	同上	
12	連曲モード	9990円	コンテンツ	10000曲		指定金額で課金される	同上	
13	なし	9990円	楽曲	6曲		課金されない	同上	
14	単曲モード	0円	コンテンツ	9999曲		指定金額で課金される	同上	
15	なし	-10円	コンテンツ	6曲		設定できない	同上	
16	単曲モード	10000円	楽曲	0曲		設定できない	同上	
17	単曲モード	10000円	コンテンツ	9999曲		設定できない	同上	
18	単曲モード	-10円	コンテンツ	0曲		設定できない	同上	
19	単曲モード	9990円	コンテンツ	10000曲		指定金額で課金される	同上	
20	なし	0円	コンテンツ	7曲		課金されない	同上	
21	単曲モード	-10円	楽曲	9999曲		設定できない	同上	
22	なし	-10円	コンテンツ	10000曲		設定できない	同上	
23	連曲モード	0円	楽曲	6曲		指定金額で課金される	同上	
24	連曲モード	-10円	コンテンツ	9曲		設定できない	同上	
25	単曲モード	0円	楽曲	1曲		指定金額で課金される	同上	
26	なし	9990円	コンテンツ	1曲		課金されない	同上	
27	単曲モード	9990円	コンテンツ	6曲		指定金額で課金される	同上	
28	なし	0円	楽曲	10000曲		課金されない	同上	
29	なし	9990円	楽曲	7曲		課金されない	同上	

…

テストケース

リスク•テスト全体を俯瞰

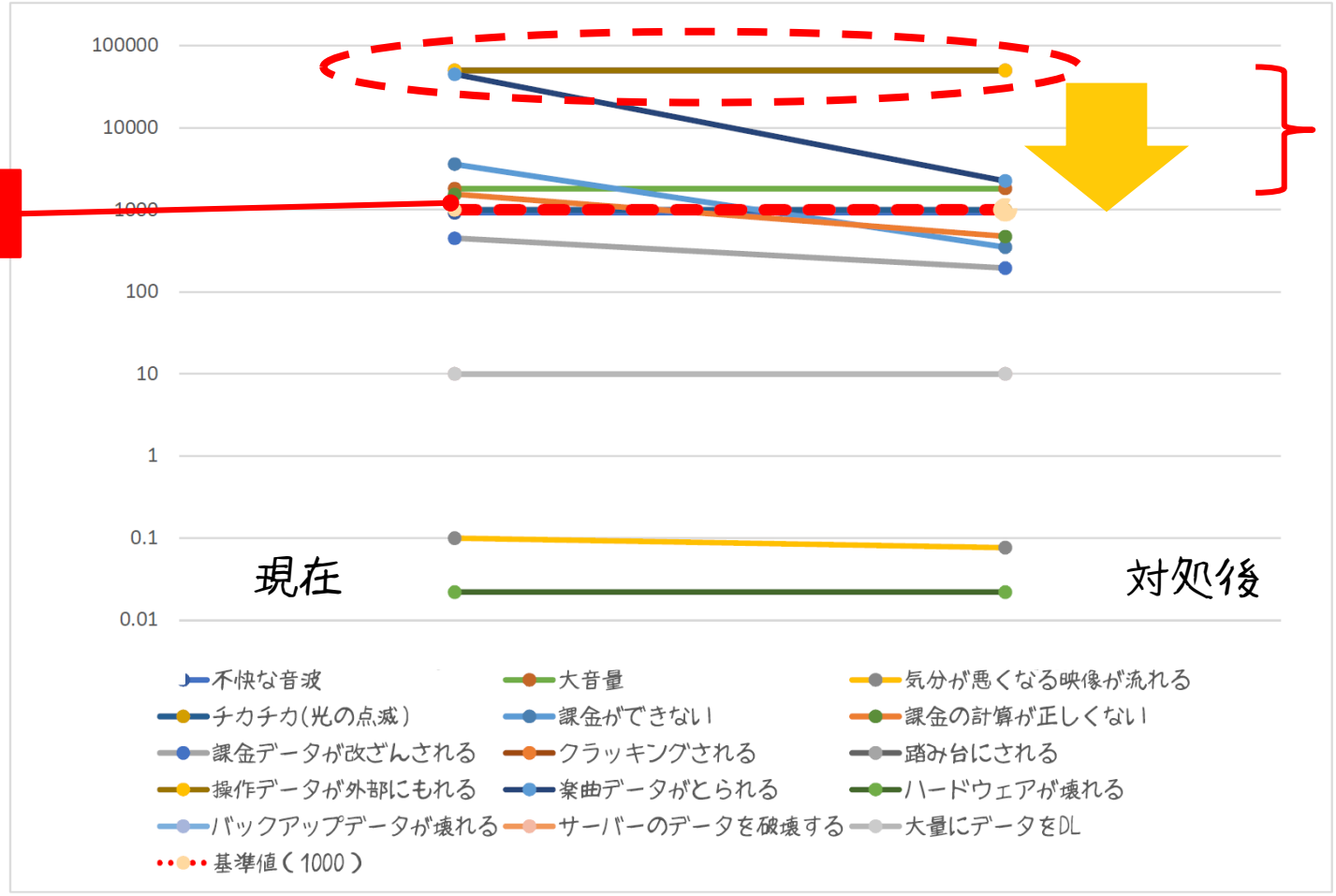
「やばさ」に対して、許容範囲になっているか、何をテストするかを俯瞰する。

				やばさ							テストタイプ				
カテゴリ	リスクカテゴリ	No.	フォールト項目	Rank	現在	⇒	Rank		低減数	低減率	データ	流れ	条件	イベント/タイミング	構成
顕在化しにくい	身体的リスク	1	不快な音波	9	917.8	⇒	7	917.8	0	0.00%	ー	ー	ー	ー	ー
		2	大音量	6	1809.32	⇒	5	1809.32	0	0.00%	ー	ー	ー	ー	ー
		3	気分が悪くなる映像が流れる	14	0.10	⇒	14	0.10	0	0.00%	ー	ー	ー	ー	ー
		4	チカチカ(光の点滅)	8	1000	⇒	6	1000	0	0.00%	ー	ー	ー	ー	ー
	金銭的リスク(課金)	14	課金ができない	5	3600.00	⇒	9	352.08	3247.92	90.22%	○	○	○	○	ー
		15	課金の計算が正しくない	7	1544.58	⇒	8	472.5	1072.08	69.41%	ー	ー	ー	○	○
		16	課金データが改ざんされる	10	450.00	⇒	10	194.58	255.42	56.76%	ー	○	ー	ー	ー
	社会的要請	17	クラッキングされる	1	50000	⇒	1	50000	0	0.00%	ー	ー	ー	ー	ー
		18	踏み台にされる	1	50000	⇒	1	50000	0	0.00%	ー	ー	ー	ー	ー
		19	操作データが外部にもれる	1	50000	⇒	1	50000	0	0.00%	ー	ー	ー	ー	ー
		20	楽曲データがとられる	4	45000	⇒	4	2250	42750	95.00%	ー	ー	ー	ー	ー
	物理的破壊リスク	21	ハードウェアが壊れる	15	0.022	⇒	15	0.022	0	0.00%	ー	ー	ー	ー	ー
		22	バックアップデータが壊れる	11	10	⇒	11	10	0	0.00%	ー	ー	ー	ー	ー
		23	サーバーのデータを破壊する	11	10	⇒	11	10	0	0.00%	ー	ー	ー	ー	ー
		24	大量にデータをDL	11	10	⇒	11	10	0	0.00%	ー	ー	ー	ー	ー
顕在化しやすい	うまく使えないリスク	5	ハウリング	ー	ー	⇒		ー			ー	ー	ー	ー	ー
		6	音が出ない	ー	ー	⇒		ー			ー	ー	ー	ー	ー
		7	映像が出ない	ー	ー	⇒		ー			ー	ー	ー	ー	ー
		8	字幕がずれる	ー	ー	⇒		ー			ー	ー	ー	ー	ー
		9	途中で止まる	ー	ー	⇒		ー			ー	ー	ー	ー	ー
		10	テンポが変わる	ー	ー	⇒		ー			ー	ー	ー	ー	ー
		11	字幕が切れる	ー	ー	⇒		ー			ー	ー	ー	ー	ー
		12	となりの音が割り込む	ー	ー	⇒		ー			ー	ー	ー	ー	ー
		13	起動しない	ー	ー	⇒		ー			ー	ー	ー	ー	ー
					204351.82	⇒		157026.4	47325.4	23.16%					

リスク•テスト全体を俯瞰

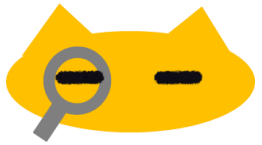
「やばさ」がどうなるかをそのリリース毎に確認していく。

基準値:1000



対応必須なもの

実現できそうか？



テスト開発コンセプト

「製品のリリースを止めるバグを許容範囲まで減らす！」

課題

1. 製品のリリースを止める事象およびバグをどう特定するか？
   ⇒
2. テストで対処する/しない事象はどのように決めるか？
   ⇒
3. 対処するためにどんなテストをどれくらい行うか？
   ⇒

実現できそうか？

テスト開発コンセプト

「製品のリリースを止めるバグを許容範囲まで減らす！」

課題

1. 製品のリリースを止める事象およびバグをどう特定するか？
   ⇒フォールトツリー図を用いた事象の分解
2. テストで対処する/しない事象はどのように決めるか？
   ⇒
3. 対処するためにどんなテストをどれくらい行うか？
   ⇒

実現できそうか？

テスト開発コンセプト

「製品のリリースを止めるバグを許容範囲まで減らす！」

課題

1. 製品のリリースを止める事象およびバグをどう特定するか？
   ⇒フォールトツリー図を用いた事象の分解
2. テストで対処する/しない事象はどのように決めるか？
   ⇒被害金額とコード変更履歴からのリスク値による優先順位
3. 対処するためにどんなテストをどれくらい行うか？
   ⇒

実現できそうか？

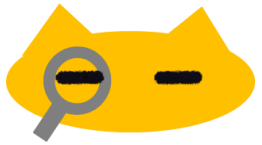
テスト開発コンセプト

「製品のリリースを止めるバグを許容範囲まで減らす！」

課題

1. 製品のリリースを止める事象およびバグをどう特定するか？
   ⇒フォールトツリー図を用いた事象の分解
2. テストで対処する/しない事象はどのように決めるか？
   ⇒被害金額とコード変更履歴からのリスク値による優先順位
3. 対処するためにどんなテストをどれくらい行うか？
   ⇒テストタイプのリスク低減表による事象の許容範囲への調節

実現できそうか？



テスト開発コンセプト

「製品のリリースを止めるバグを許容範囲まで減らす！」

課題

1. 製品のリリースを止める事象およびバグをどう特定するか？
   ⇒フォールトツリー図を用いた事象の分解
2. テストで対処する/しない事象はどのように決めるか？
   ⇒被害金額とコード変更履歴からのリスク値による優先順位
3. 対処するためにどんなテストをどれくらい行うか？
   ⇒テストタイプのリスク低減表による事象の許容範囲への調節

以上で、本提案のご説明を終わります。
ご清聴ありがとうございました。