# 『スープカレー方式』による システムテスト分析と設計

## －地域有志勉強会によるプロセス実践･解析の成果発表－

TEF北海道テスト勉強会（TEF-道）

How Do you like software test?


TEF北海道テスト勉強会
TEF-DO =Testing Engineer's Forum in hokkaiDO

【TEF-道】
- 上田 和樹
- 小楠 聡美
- 高木 進也（東京エレクトロン ソフトウェアテクノロジーズ(株)）
- 中岫 信（東京エレクトロン ソフトウェアテクノロジーズ(株)）
- 根本 紀之（東京エレクトロン ソフトウェアテクノロジーズ(株)）
- 藤田 将志（東京エレクトロン ソフトウェアテクノロジーズ(株)）
- 古川裕一郎
- 村上 寛
- 安達 賢二(お世話係)

# アジェンダ

- TEF北海道テスト勉強会の紹介
- テスト分析と設計
- 5W1H
- マインドマップ
- スープカレー方式
- HAYST法
- まとめ

TEF-道

How do you like software test?

# TEF北海道テスト勉強会の紹介-1

- 「TEF札幌テスト勉強会」から改名
  - 「JaSST北海道」に合わせてみました。
- 隔週で一回2時間程度の開催。参加メンバーは10名程度で、常時5～6名のメンバーが参加。
- 愛称「TEF-道（てふ-どう）」
  - 北海「道」
  - テスト「道」
  - DO・・・ 行う, する, なしとげる, 終える
  - テスト「どう」でしょう？

TEF-道

How do you like software test?

# TEF北海道テスト勉強会の紹介-2

- 2006～2008
  - JSTQBテスト勉強会として発足
  - その後TEF札幌テスト勉強会として断続的に活動を行う
- 2009年1月～6月（第1ステージ）
  - 架空のペットボトル式加湿器の仕様書を使用して、テストプロセスの実践と解析を継続的に行う
- 2009年7月～9月（第2ステージ）
  - 架空のモバイル端末上のアプリケーションの仕様書をもとにして、再度テストプロセスの実践と解析を行う
  - JaSST‘09北海道にて成果発表
- 2009年10月～12月（第3ステージ）
  - その後、発表内容をさらに勉強して理解を深める
  - JaSST‘10東京にて事例発表

TEF-道

How do you like software test?

# ここから本題

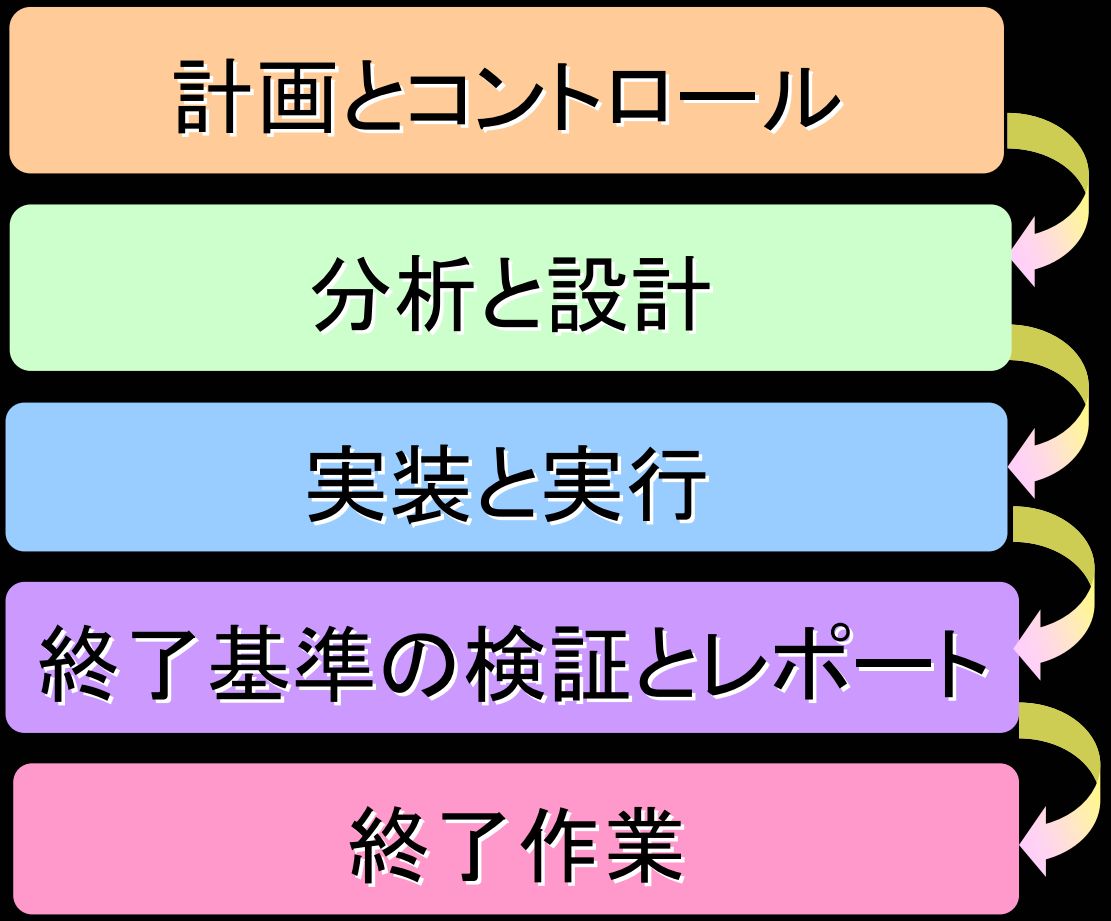
TEF-道

How do you like software test?

まずは
テスト
フェーズ
を考えた

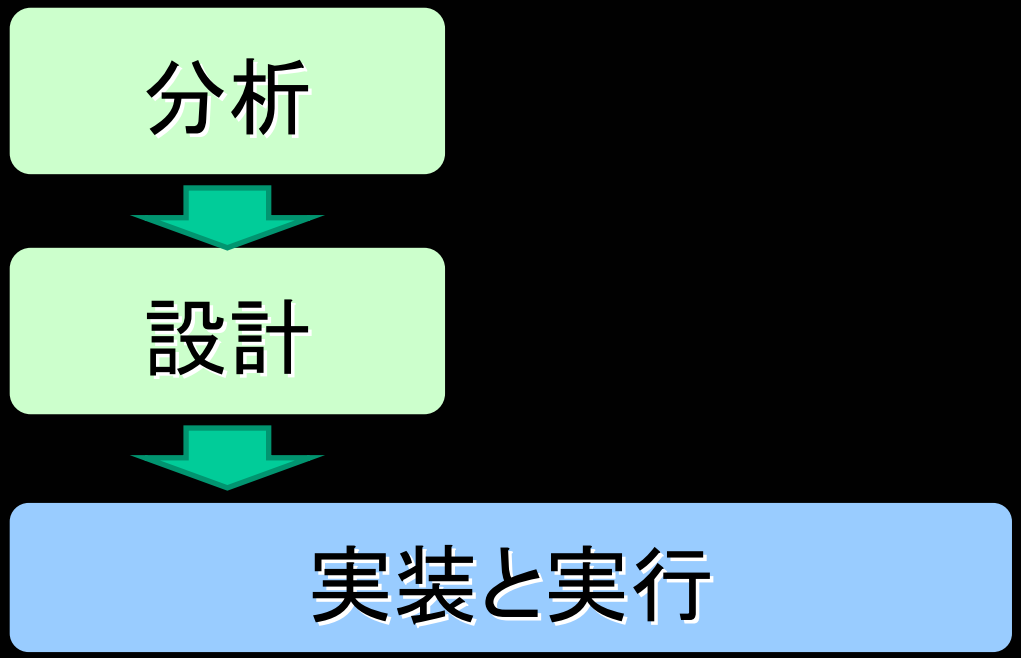
TEF-道
How do you like software test?

# 基本的なテストプロセス （JSTQBより）




TEF-道
How do you like software test?

# 以下の３点にフォーカスを絞った




TEF-道
How do you like software test?

# テストフェーズ

- 「テスト分析」フェーズとは
  『思考を発散させる』(風呂敷を広げる)こと
  ⇒三色ボールペンでの仕様書の読み込みやマインドマップにて観点を広げていく

- 「テスト設計」フェーズとは
  『思考をまとめる』(風呂敷をたたむ)こと
  ⇒マインドマップを整理したり、表にまとめたりする

- 「テスト実装と実行」フェーズとは
  『テスト仕様』の作成
  ⇒「手順」「条件」「期待値」が備わっている「テストケース」の集合体

TEF-道
How do you like software test?

# システムテストを実践してみた

TEF-道

How do you like software test?

しかも
テスト技法
を駆使する
壮大な計画

TEF-道
How do you like software test?

# 『テストプロセスの実践と解析』

## 分析と設計

どの観点で書けばよい？

この間をうまくつなげられなかった・・

- 製品のウリや要求品質などを考慮
- 三色ボールペンを使用して仕様読込
- マインドマップを使用してテスト観点の切り出
- HAYST法を使用してテスト設計
- ペルソナ法によるユーザーシナリオ作成
- リスクベース法によるテストの優先付け

TEF-道

How do you like software test?

# 反省した

TEF-道

How do you like software test?

# 題材：超シンプルメモ帳

無題 － Simple メモ帳 ＿ □ ×

ファイル　　書式

■シンプルメモ帳仕様
□ファイル
①新規作成
新しいテキストファイルを開きます
②開く
既存のテキストファイルを選択して、当プログラムで開きます
③上書き保存
現在開いているファイルに上書きして保存します
④名前を付けて保存
別ファイルとして名前を付けて保存します
⑤ページ設定
ページの設定を行います
⑥印刷設定
印刷設定を行います
⑦終了
プログラムを終了します
□書式
①フォント
テキストフォントの設定を行います

分析の

切り口

を考えた

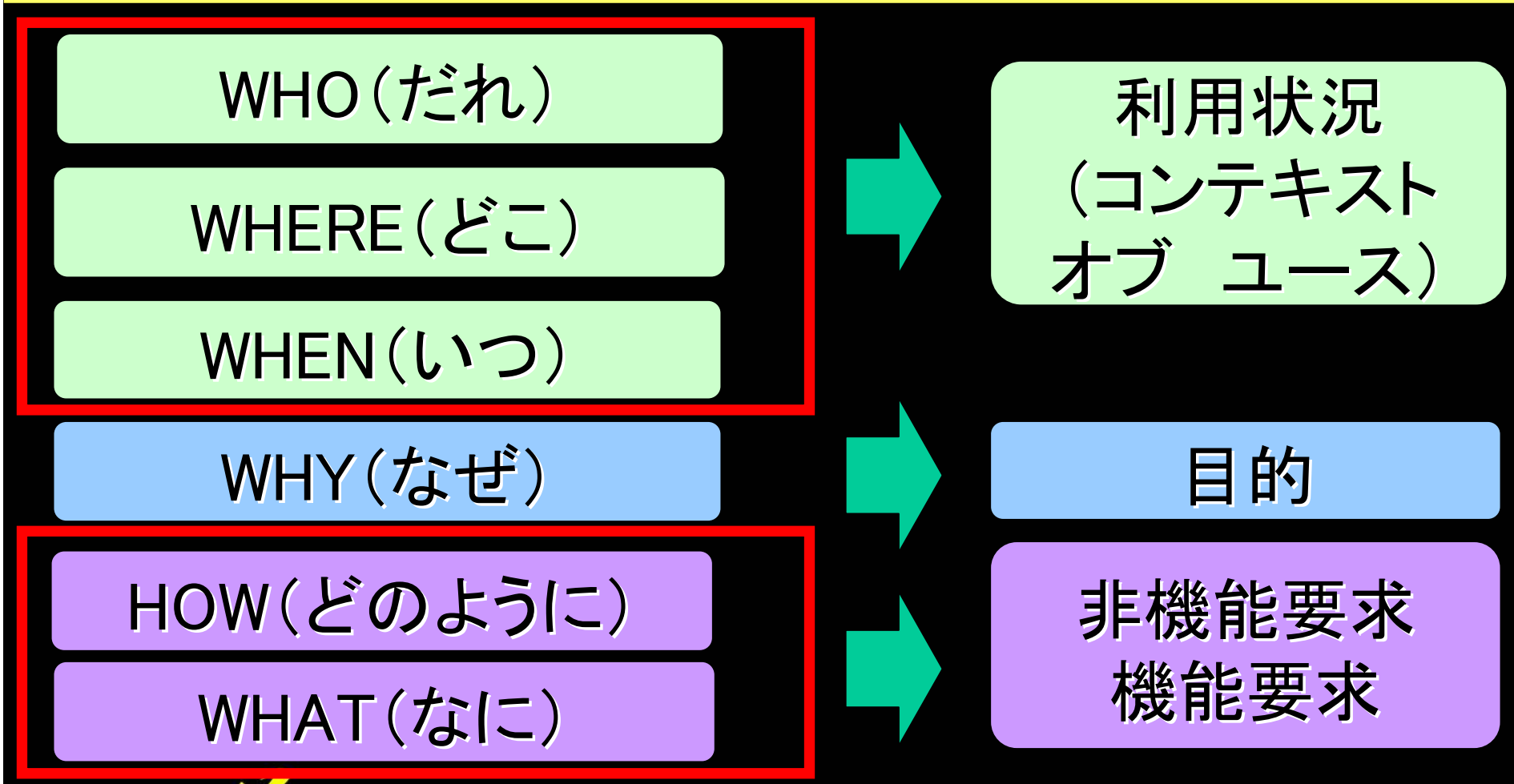
TEF-道

How do you like software test?

# システムを「5W1H」で考えてみる

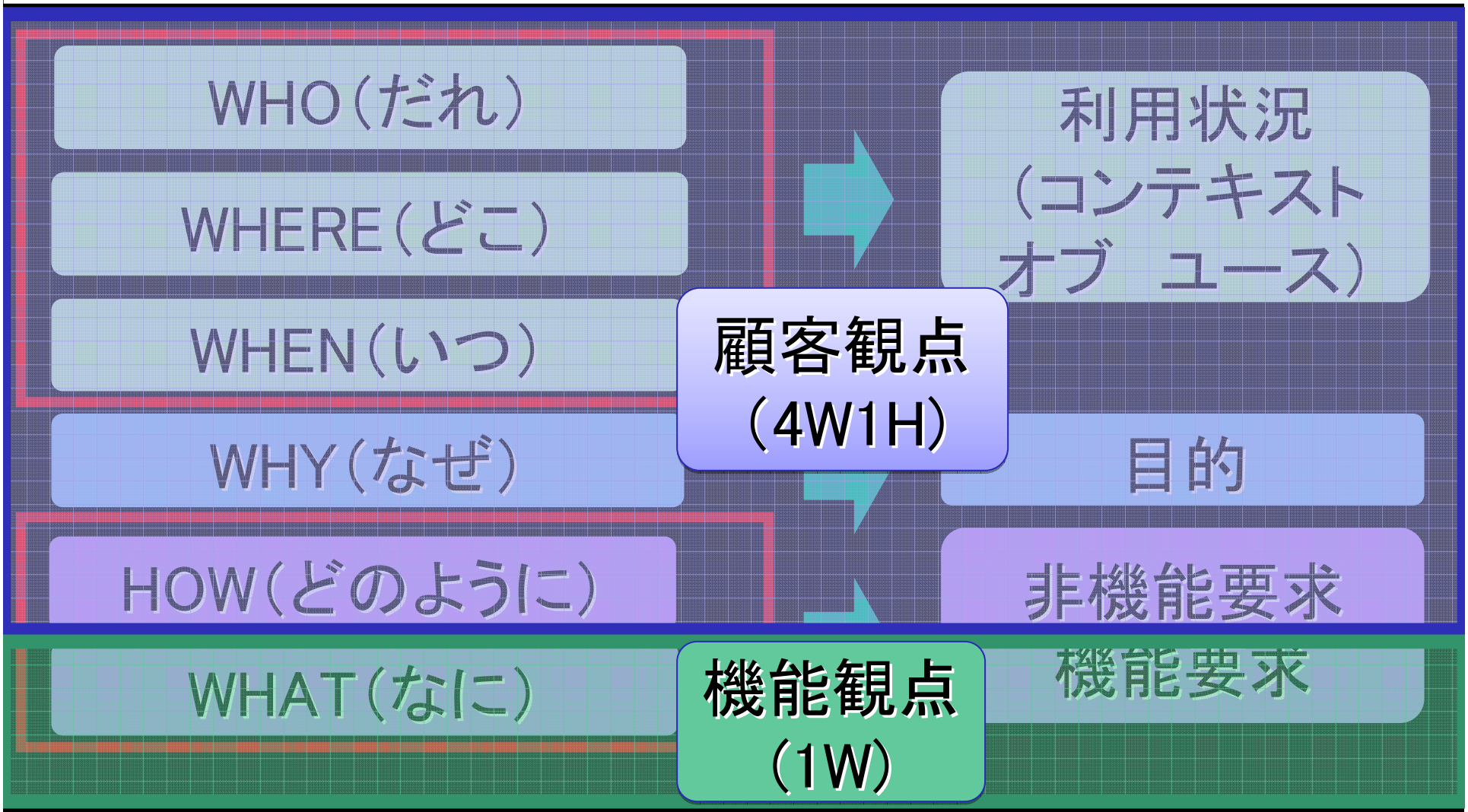
| | 内容と事例 | 意味 |
|---|---|---|
| WHO | 誰が、どのような人が利用するのか<br>例：開発者が | 利用者・ユーザー |
| WHERE | どこで利用するのか<br>例：会議室で | 利用場所・場面 |
| WHEN | いつ利用するのか<br>例：会議中に | 利用タイミング・場面 |
| WHY | どうして必要なのか<br>例：手軽に情報を保管し、活用したい | システムの目的・目標 |
| HOW | どのように使えるとよいのか<br>例：すばやく、多くの機種で | 非機能要求 |
| WHAT | どんな機能を使うのか<br>例：メモを取り書き込む | 機能要求 |

# 「5W1H」をグループ分けする




TEF-道
How do you like software test?

# 「5W1H」をグループ分けする

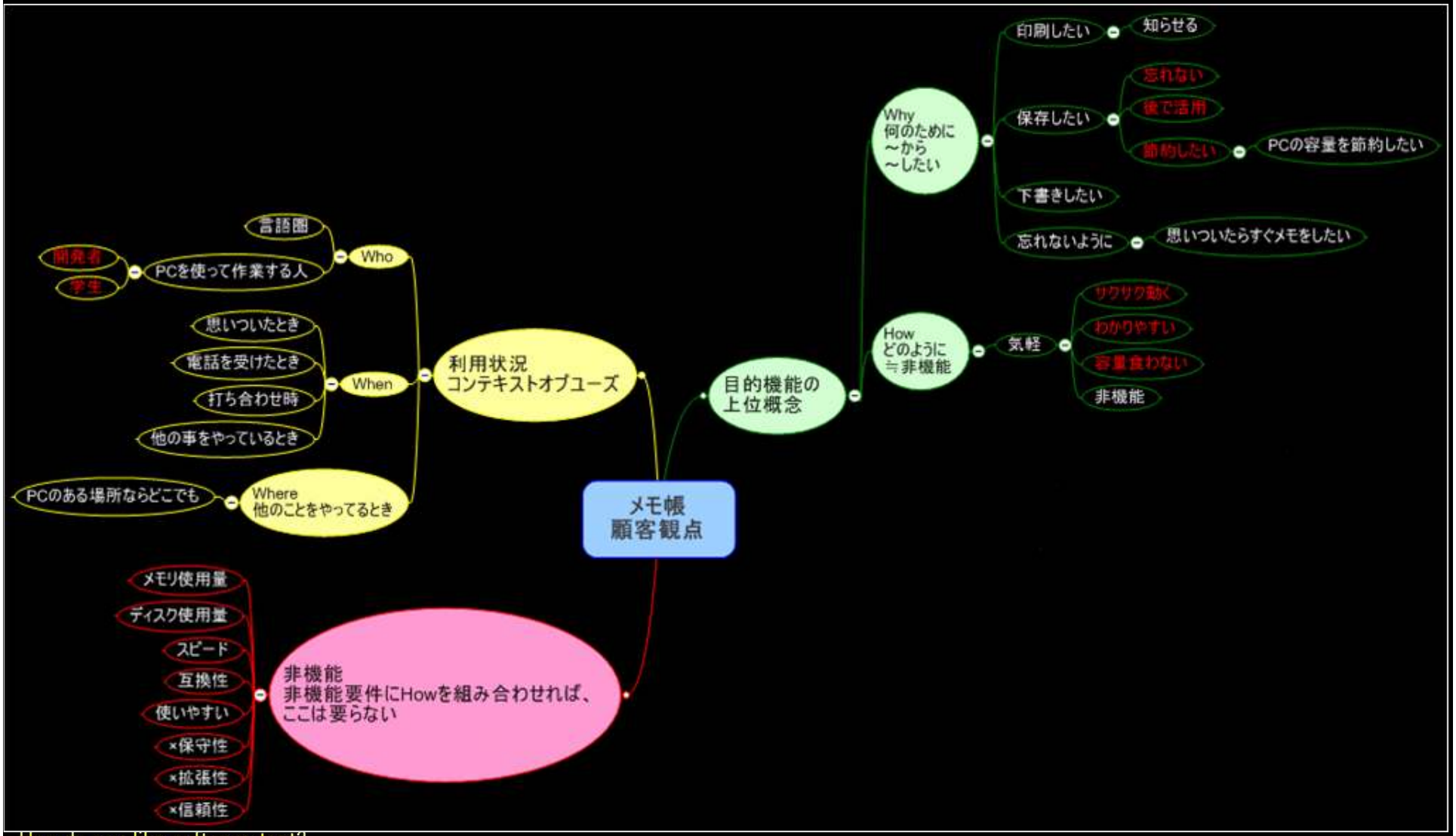


4W1H + 1W

でマインドマップ

を考えた

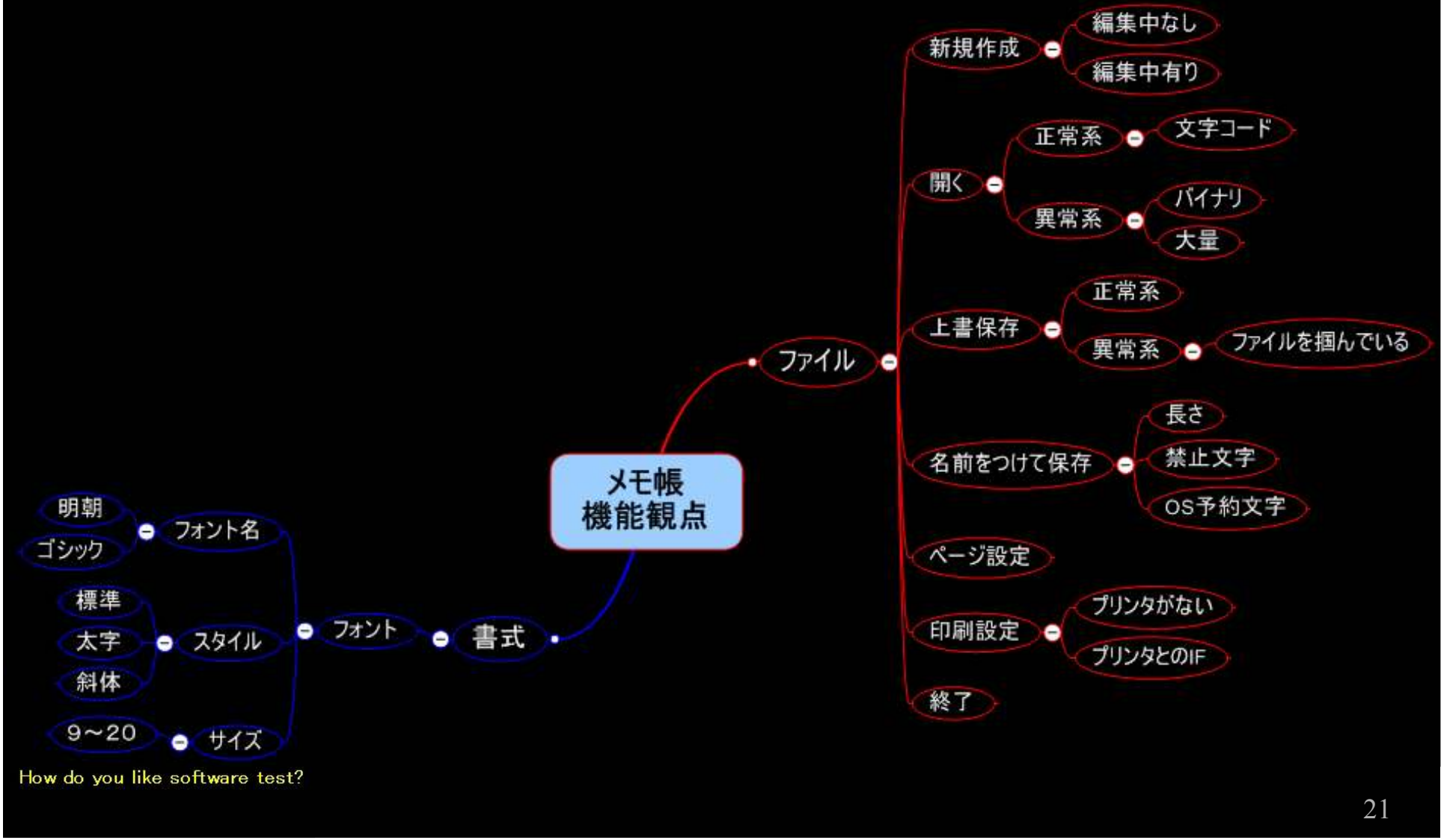
# 4W1H･･･顧客観点のマインドマップ



# 顧客観点マインドマップのポイント

- 4W1Hで考える
  - Why ⇒目的
    - そのシステムの目的
  - When+Where+Who⇒利用状況（コンテキストオブ　ユース）
    - そのシステムが使われる状況
      - 開発者が仕事で使う／PC初心者が自宅で使う　etc…
  - How ⇒非機能要件
    - どのように動作するか？
      - 素早く／使いやすく　　etc…

# 1W(WHAT)・・・機能観点のマインドマップ



# 機能観点マインドマップのポイント

- 機能を網羅する
  - 「機能」とは、顧客が何かしらの『目的』を達成するものとする
- 「顧客観点」では浮かばない発想を広げる
- 正常系と異常系を考える

TEF-道
How do you like software test?

二枚のマップを融合できないか？

TEF-道

How do you like software test?

ここで

スープ

カレー方式

登場

TEF-道

How do you like software test?

# スープカレーとは・・・

•北海道を代表するローカルフード
•一般的に、ごはんをスプーンですくって、スープ状のカレーにくぐらせて、スープをごはんに「絡ませて」食べる。

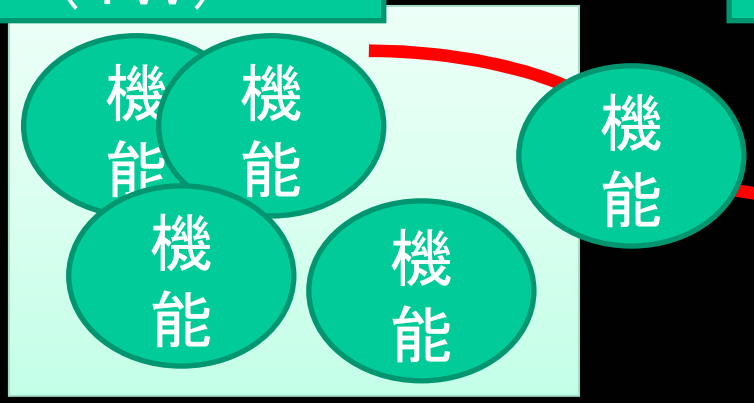
# スープカレー方式のやりかた

ごはん＝機能観点
カレー＝顧客観点　　・・・と見立てて

「機能のごはん」をひとつずつ、「顧客のスープ」にくぐらせるイメージ

# 顧客観点という名のスープにくぐらすと・・・？

機能のごはん
（1W）

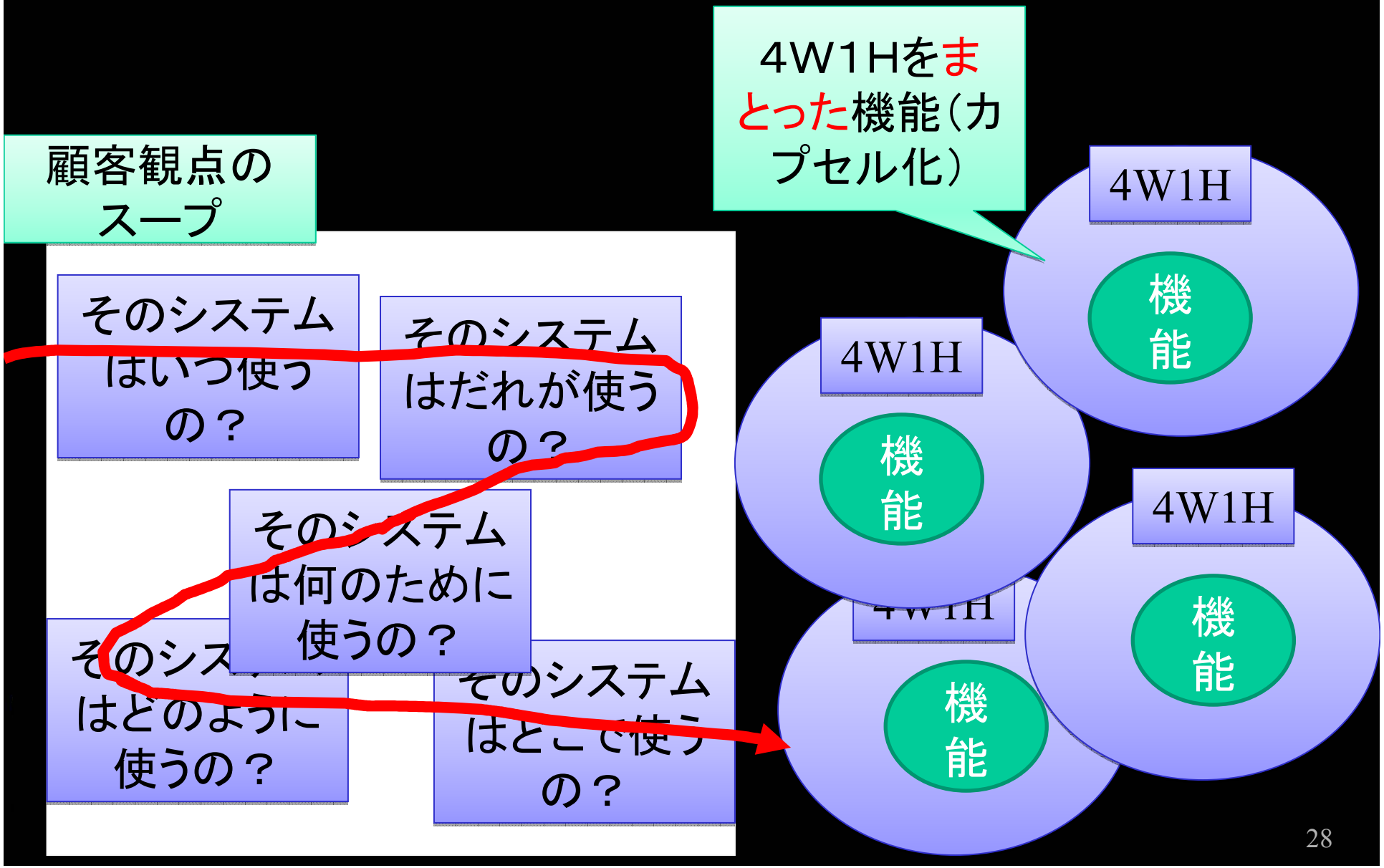



顧客観点のスープ
（4W1H）

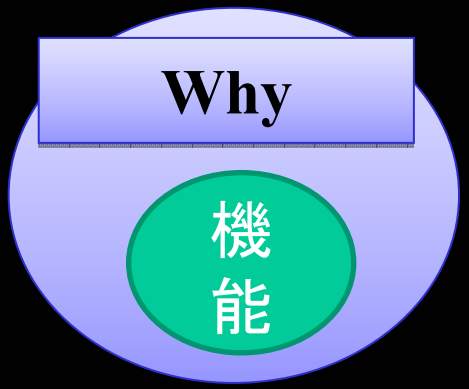
- そのシステムはいつ使うの？
- そのシステムはだれが使うの？
- そのシステムは何のために使うの？
- そのシステムはどのように使うの？
- そのシステムはどこで使うの？

TEF-道
How do you like software test?

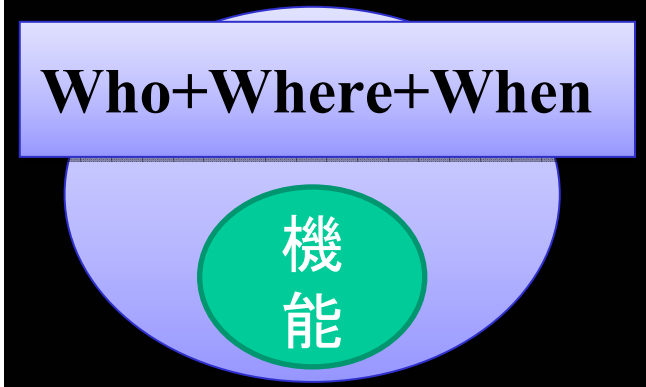
# 4W1Hをまとった「機能たち」ができる



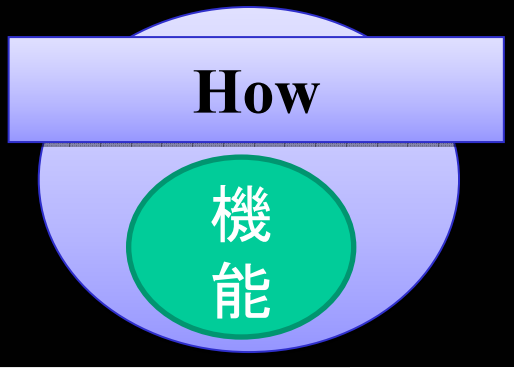
# こいつらの正体は？




その機能はどのような目的で存在しているか？（目的機能）




その機能が特定の利用状況で顧客満足を満たせるか？




その機能がどのように動作するか？（非機能要件）

# 具体的な「混ぜる」方法

- スープカレー表を利用
- 基本は縦に「機能観点」、横に「顧客観点」
- その「機能観点」×「顧客観点」の交点を埋めていく

| | 顧客観点 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 目的 (Why) | | | コンテキスト (When+Where+Who) | | | 非機能項目 (How) | | |
| 機能観点 (What) | 目的1 | 目的2 | 目的3 | コンテキスト1 | コンテキスト2 | コンテキスト3 | 非機能項目1 | 非機能項目2 | 非機能項目3 |
| 機能1 | | | | | | | | | |
| 機能2 | | | | | | | | | |
| 機能3 | | | | | | | | | |
| 機能4 | | | | | | | | | |

交点を埋めていく

# 設計観点を加えた例

- 機能観点マインドマップで発想した項目を横軸に記入
  - 異常系は縦軸に配置できない
  - ある機能の発想項目を他の機能に適用できる

| | 顧客観点 | | | | | | | | | 設計観点 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 目的（Why） | | | コンテキスト（When+Where+Who） | | | 非機能項目（How） | | | 異常系／機能観点で気が付くこと | | | |
| 機能観点（What） | 目的1 | 目的2 | 目的3 | コンテキスト1 | コンテキスト2 | コンテキスト3 | 非機能項目1 | 非機能項目2 | 非機能項目3 | 最大値 | バイナリ | 0データ | 同時アクセス |
| 機能1 | | | | | | | | | | | | | |
| 機能2 | | | | | | | | | | | | | |
| 機能3 | | | | | | | | | | | | | |
| 機能4 | | | | | | | | | | | | | |

# 想定ユーザーと想定外ユーザーを分けた例

- 利用者には「想定ユーザー」と「想定外ユーザー」がある

| | 顧客観点 | | | | | | | | | 設計観点 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 目的(Why) | | | 利用者(Who) | | | 利用状況(When+Where) | | | 非機能項目(How) | | | 異常系／機能観点で気が付くこと |
| 機能観点(What) | 目的1 | 目的1 | 目的1 | 想定利用状況 | 想定利用状況 | 想定外利用状況 | 想定外利用状況 | 非機能項目1 | 非機能項目1 | 非機能項目1 | 最大値 | バイナリ | 0データ | 同時アクセス |
| 機能1 | | | | | | | | | | | | | | |
| 機能2 | | | | | | | | | | | | | | |
| 機能3 | | | | | | | | | | | | | | |

# スープカレー表の例-1

| | | 顧客観点 | | | | | | | | | | | | | 機能観点 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 【WHAT】機能要求 | | 【WHY】システム全体の目的 | | | 【WHO】利用者 | | | | 【HOW】非機能要求 | | | | | | | | |
| 大項目 | 中項目 | 文章を印刷したい | 文章を保存して次回も使用したい | テキストファイルを閲覧／編集したい | 想定ユーザー | | | 利用する可能性のある想定外ユーザー | ①レスポンスが早い | ②ファイル容量が少ない | ③大抵のPCで使える | ④類似アプリでも使える | ⑤基礎ITリテラシでも使える | ⑥利用方法が類推しやすい | ①バイナリファイルの扱い | ②大量データの扱い | 3バイト文字の扱い |
| | | | | | 開発者 | | ビジネスマン | パソコン初心者 | | | | | | | | | |
| | | | | | 【WHEN+WHERE】いつ・どこで | | | | | | | | | | | | |
| | | | | | ソースを見たい | 設定ファイルを編集する | TODOリストを書く | 何気なく | | | | | | | | | |
| アプリ操作 | 起動 | | | 起動後、すぐにテキスト入力が可能 | すぐに起動できる。 | すぐに起動できる。 | すぐに起動できる。 | 起動のアイコンが「スタート」⇒「プログラム」の中にあって、そこから起動できる。 | ○（1～2秒） | × | ○ | × | ○（説明なしでも起動できる） | ○（説明なしで迷わず操作できる） | | | |
| ファイル操作 | 新規作成 | | | ・新規作成後、テキスト編集が行なえる<br>・編集中のファイルがある場合は、保存確認のダイアログが表示される | 起動後にすぐ新規作成が行なえる | 起動後にすぐ新規作成が行なえる | 起動後にすぐ新規作成が行なえる | 「新規作成」が初心者でも行なえる | ○（1～2秒） | × | ○ | × | ○（説明なしでも新規で作成できる） | ○（説明なしで迷わず操作できる） | | | |
| ファイル操作 | 開く | 開いたテキストを印刷できる | 保存したファイルを開くことができる | ・既存ファイルを開いて閲覧可能<br>・既存ファイルに対して編集が可能 | 「.c」などの拡張子が開ける | iniファイルが開けて編集できる | 前回保存ファイルのパスがデフォルト表示される | 「前回編集したファイル」を初心者でも開くことができる | △ | × | ○ | × | 説明なしでも既存ファイルを読み込める | 説明なしで迷わず操作できる） | テキストとして開く | 最大値を超えるデータは編集不可 | ・指定文字コード以外表示不可にする<br>・3バイトの漢字を含むファイル（UNIコード）を開く。 |
| 環境設定 | 頁設定 | 印刷設定の通りに印刷できる | ページ設定も保存されて、次回も同じ設定で使用できる<br>※設定多数あり | 設定した印刷設定にて編集可能<br>※設定多数あり | | | | | △ | × | ○ | × | ○（説明なしでも頁設定ができる） | ○（説明なしで迷わず操作できる） | | | |

# スープカレー表の例-1

機能の「目的」が洗い出せる

多様な利用状況に対応した機能の確認

機能が満たすべき「非機能要件」が洗い出せる

機能観点から洗い出した異常系を洗い出せる

| 【WHAT】機能要求 | | 顧客観点 | | | | | | | | | | 機能観点 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 【WHY】システム全体の目的 | | | 【WHO】利用者 | | | | 【HOW】非機能要求 | | | | | |
| 大項目 | | | | | ソースを見たい | ファイルを編集する | TODOリストを書く | 何気なく | ①レスポンスが早い | | ⑥利用方法が類しやす | ①バイナリ | ②大量 | 3バイト文 |
| アプリ操作 | 起動 | | | 起動後、すぐにテキスト入力が可能 | すぐに起動できる。 | すぐに起動できる。 | すぐに起動できる。 | 起動のアイコンが「スタート」⇒「プログラム」の中にあって、そこから起動できる。 | （1～2秒） | × | ○（説明なしでも起動できる） | ○（説明なしで迷わず操作できる） | | | |
| ファイル操作 | 新規作成 | | | ・新規作成後、テキスト編集が行なえる<br>・編集中のファイルがある場合は、保存確認のダイアログが表示される | 起動後にすぐ新規作成が行なえる | 起動後にすぐ新規作成が行なえる | 起動後にすぐ新規作成が行なえる | 「新規作成」が初心者でも行なえる | （1～2秒） | × | ○（説明なしでも新規で作成できる） | ○（説明なしで迷わず操作できる） | | | |
| ファイル操作 | 開く | 開いたテキストを印刷できる | 保存したファイルを開くことができる | ・既存ファイルを開いて閲覧可能<br>・既存ファイルに対して編集が可能 | 「.c」などの拡張子が開ける | iniファイルが開けて編集できる | 前回保存ファイルのパスがデフォルト表示される | 「前回編集したファイル」を初心者でも開くことができる | 4 | × | ○ | × | 説明なしでも既存ファイルを読み込める | 説明なしで迷わず操作できる） | テキストとして開く | 最大値を超えるデータは編集不可 | ・指定文字コード以外表示不可にする<br>・3バイトの漢字を含むファイル（UNIコード）を開く。 |
| 環境設定 | 頁設定 | 印刷設定の通りに印刷できる | ページ設定も保存されて、次回も同じ設定で使用できる<br>※設定多数あり | 設定した印刷設定にて編集可能<br>※設定多数あり | | | | | 4 | × | ○ | × | ○（説明なしでも頁設定ができる） | ○（説明なしで迷わず操作できる） | | | |

# スープカレー表の例-2

| 【WHY】システム全体の目的達成確認 | |
|---|---|
| 新規作成・編集・保存（印刷）<br>残しておきたい情報を見つけた（あるいは思いついた）時、手軽に書き留め（または貼り付け）、あるいは編集し、すぐに印刷または保存したい | 既存ファイル編集・保存（印刷）<br>保存していた情報を手早く再利用（呼び出して印刷・編集・保存等）したい |

システム全体の「目的」をまとめて明示する

| 顧客観点 | | | | | | | | | | | | 機能観点 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 【WHAT | | 者 | | | | 【HOW】非機能要求 | | | | | | | | |
| 大項 | | | | | 利用する可能性のある想定外ユーザー | | | | | | | | | |
| | | | | ビジネスマン | 初心者 | | | | | | | | | |
| | | 【WHE】いつ・どこで | | | | | | | | | | | | |
| | | ソースを見たい | 設定ファイルを編集する | TODOリストを書く | 何気なく | ①レスポンスが早い | ②ファイル容量が少ない | ③大抵のPCで使える | ④類似アプリでも使える | ⑤基礎ITリテラシでも使える | ⑥利用方法が類推しやすい | ①バイナリファイルの扱い | ②大量データの扱い | 3バイト文字の扱い |
| 一般ST | | | | | | ST1 | ST1 | ST1 | | | | ST1 | ST1 | ST1 |
| 別PCテスト | | | | | | ST2 | ST2 | ST2 | | | | | | |
| 簡易シナリオテスト | | | | | | | | | | ST3 | ST3 | | | |
| アプリ操作 | 起動 | すぐに起動できる。 | すぐに起動できる。 | すぐに起動できる。 | 起動のアイコンが「スタート」⇒「プログラム」の中にあって | ○（1～2秒） | | ○ | × | ○（説明なしでも起動できる） | ○（説明なしで迷わず操作できる） | | | |
| ファイル操作 | 新規作成 | 起動後にすぐ新規作成が行なえる | 起動後にすぐ新規作成が行なえる | 起動後にすぐ新規作成が行なえる | | | | ○ | × | ○（説明なしでも新規で作成できる） | ○（説明なしで迷わず操作できる） | | | |
| ファイル操作 | 開く | 「.c」などの拡張子が開ける | iniファイルが開けて編集できる | 前回保存ファイルのパスがデフォルト表示される | 「前回編集したファイル」を初心者でも開くことができる | ○（1～2秒） | | ○ | ○別アプリoutファイルを開く（そして編集等を行う） | 説明なしでも既存ファイルを読み込める | 説明なしで迷わず操作できる） | テキストとして開く | 最大値を超えるデータは編集不可 | ・指定文字コード以外表示不可にする<br>・3バイトの漢字を含むファイル（UNIコード）を開く。 |

システムテストでの確認方法を記述

これによって
何が
できるのか？

TEF-道
How do you like software test?

# たとえば

# HAYST法

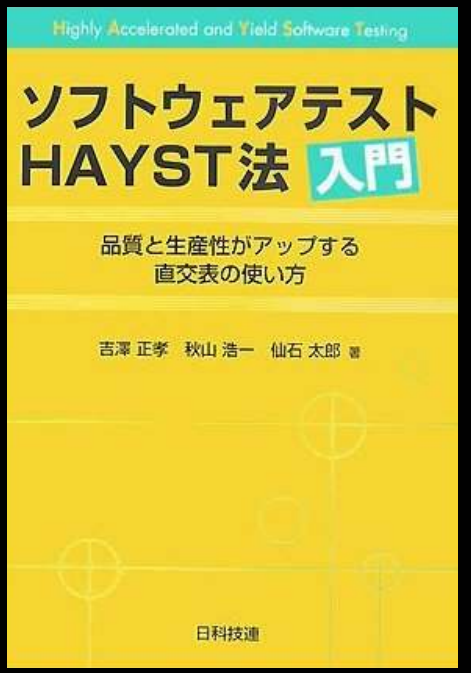
TEF-道

How do you like software test?

# HAYST法とは？

- 直交表を使ったテストプロセスの実施方法
  - 直交表とは・・・？
    - 「ペアテスト」の一種。
    - 「全組合せテスト」の代わりに「二項目間の組合わせ」を網羅するテスト
    - 品質リスクをそれほど落とさずに、テストケースを削減できる
  - FV表とFL表を使用して因子と水準を洗い出して、組合せテストを行なう

→因子と水準の洗い出しが非常に重要！



# FV表（機能検証条件表）及びFL表（因子水準表）

| Function-Verification Table | | | | Factor-Level Table | |
|---|---|---|---|---|---|
| No. | Function（目的機能） | Verification（検証方法） | Test（テスト技法） | Factor(因子) | Level(水準) |
| 1 | 起動 | | | | |
| 1.1 | 手軽に、すぐに入力するために、起動開始からすぐに立ち上がる | ・プログラムファイルから起動して2秒以内で画面が表示されることを確認<br>・プログラム名を直接入力して2秒以内で画面が表示されることを確認<br>・スタートファイルの「プログラム」のファイル名とアイコンが分かりやすいことを確認 | | | |
| 1.2 | 手軽に、すぐに入力するために、起動後そのままテキスト入力が可能になる | ・起動後、フォーカスがメイン画面に当たっていることを確認<br>・キーボードを入力して、すぐに文字入力が可能であることを確認 | | | |
| 2 | 新規作成 | | | | |
| 2.1 | 手軽に、すぐに、そして説明書なしでも簡単に類推して使えるように、操作が分かりやすく行なえ、手続き後すぐに新規作成可能になる | ・上記起動後そのまま新規作成できる状態になることを確認→そのことを認識して自然と入力開始することを確認<br>・「新規作成」を実行してから1～2秒以内に新しいファイルが入力可能状態で表示されることを確認<br>・編集中ファイルがある場合は、保存確認のダイアログが表示されて注意を促す | | | |
| 2.2 | 手軽に、すぐに入力するために、新規作成後に余計な操作をすることなく編集が行なえる | ・起動後、フォーカスがメイン画面に当たっていることを確認<br>・キーボードを入力して、すぐに文字入力が可能であることを確認 | | | |
| 3 | 開く | | | | |
| 3.1 | 前回使用した作業の続きを行うために、保存した既存ファイルを説明なしに分かりやすく開くことができる | ・開く操作から2秒以内にファイルが表示できることを確認<br>・前回保存ファイルのパスがデフォルト表示されることを確認 | ファイルの大きさ、種類、拡張子などを直交表で設計 | ファイルサイズ | ・100 B<br>・1 KB<br>・10 KB |
| | | | | 種類 | ・C言語ソースファイル<br>・HTMLファイル<br>・日本語テキストファイル<br>・3バイト文字を含むファイル |
| | | | | エンコード | ・JIS<br>・SHIFT-JIS |

# FV表（機能検証条件表）及びFL表（因子水準表）

| Function-Verification Table | | | | Factor-Level Table | |
|---|---|---|---|---|---|
| No. | Function（目的機能） | Verification（検証方法） | Test（テスト技法） | Factor（因子） | Level（水準） |
| 1 | 起動 | | | | |
| | 手軽に、すぐに入力するため、起動開始から | ・プログラムファイルから起動して2秒以内にが表示されることを確認<br>・プ 内<br>表示<br>・ス<br>ファイ | | | |
| | 力す<br>その<br>が可 | ・起 当た<br>こと<br>キー 入力が可能であることを確認 | | | |
| 2 | | | | | |
| 2.1 | 説明書なしでも簡単に類推して使えるように、操作が分かりやすく行なえ、手続き後すぐに新規作成可能になる | ・上記起動後そのまま新規作成できる状態になることを確認→そのことを認識して自然と入力開始することを確認<br>・「新規作成」を実行してから1～2秒以内に新しいファイルが入力可能状態で表示されることを確認<br>・編集中ファイルがある場合は、保存確認のダイアログが表示されて注意を促す | | | |
| 2.2 | 手軽に、すぐに入力するために、新規作成後に余計な操作をすることなく編集が行なえる | ・起動後、フォーカスがメイン画面に当たっていることを確認<br>・キーボードを入力して、すぐに文字入力が可能であることを確認 | | | |
| 3 | 開く | | | | |
| 3.1 | 前回使用した作業の続きを行うために、保存した既存ファイルを説明なしに分かりやすく開くことができる | ・開く操作から2秒以内にファイルが表示できることを確認<br>・前回保存ファイルのパスがデフォルト表示されることを確認 | ファイルの大きさ、種類、拡張子などを直交表で設計 | ファイルサイズ<br>種類<br>エンコード | ・100 B<br>・1 KB<br>・10 KB<br>・C言語ソースファイル<br>・HTMLファイル<br>・日本語テキストファイル<br>・3バイト文字を含むファイル<br>・JIS<br>・SHIFT-JIS |

「Why+機能」や「How+機能」によって、抽出した機能に対する目的（＝**目的機能**）の洗い出しが行なえる

目的機能を検証する「手段」を決定していく

検証「手段」を具体的にテスト実装するための「テスト技法」を決定

「検証」のためのパラメータ（因子と水準）を、機能観点（設計観点）などから抽出する

**ここを直交表で組合せ**

# 直交表の適用例

| FV項目 | No. | ファイルサイズ | 種類 | エンコード |
|---|---|---|---|---|
| 3.1<br>9.1<br>10.1 | 1 | 100 B | 日本語テキストファイル | JIS |
| | 2 | 100 B | HTMLファイル | SHIFT-JIS |
| | 3 | 100 B | C言語ソースファイル | EUC |
| | 4 | 100 B | 3バイト文字を含むファイル | UNICODE |
| | 5 | 1 KB | 日本語テキストファイル | SHIFT-JIS |
| | 6 | 1 KB | HTMLファイル | JIS |
| | 7 | 1 KB | C言語ソースファイル | UNICODE |
| | 8 | 1 KB | 3バイト文字を含むファイル | EUC |
| | 9 | 10 KB | 日本語テキストファイル | EUC |
| | 10 | 10 KB | HTMLファイル | UNICODE |
| | 11 | 10 KB | C言語ソースファイル | JIS |
| | 12 | 10 KB | 3バイト文字を含むファイル | SHIFT-JIS |
| | 13 | 100 B | 日本語テキストファイル | UNICODE |
| | 14 | 100 B | HTMLファイル | EUC |
| | 15 | 100 B | C言語ソースファイル | SHIFT-JIS |
| | 16 | 100 B | 3バイト文字を含むファイル | JIS |

# テスト仕様書の例

| FV No. | 1.1 | |
|---|---|---|
| シナリオ名 | 起動-2 | |
| 試験目的（ゴール） | OSからすばやく起動でき、すぐにテキスト入力が可能 | |
| No. | 試験手順 | 期待値 |
| 1 | 「スタート」⇒「ファイル名を指定して実行」⇒「simplememo.exe」と入力 | |
| 2 | メモ帳の初期表示がされることを確認する | ・1～2秒以内に表示されることを確認<br>・フォーカスがテキスト入力画面にあることを確認 |
| 3 | テキスト入力を行う | テキスト入力画面にテキストが表示されることを確認 |

| FV No. | 3.1 | |
|---|---|---|
| シナリオ名 | 開く-1 | |
| 試験目的（ゴール） | 保存した既存ファイルを説明なしに分かりやすく開くことができる | |
| No. | 試験手順 | 期待値 |
| 1 | 「ファイル」⇒「開く」ボタンを押下 | 「開く」ダイアログが表示される |
| 2 | テストデータの格納フォルダに移動する | |
| 3 | ファイル1を選択する　※右記テストデータ参照 | |
| 4 | 「OK」を押下 | 日本語テキストファイルが表示される |
| 5 | 「ファイル」⇒「開く」ボタンを押下 | |
| 6 | 前回開いたフォルダが表示される | |
| 7 | ファイル2を選択する | |
| 8 | 「OK」を押下 | HTMLファイルが表示される |
| | | |
| | ～【略】～ | |
| | | |
| 25 | ファイル16を選択する | |
| 26 | 「OK」を押下 | 3バイト文字を含むファイルが表示される |
| 27 | 「ファイル」⇒「閉じる」を押下する | シンプルメモ帳の画面が閉じる |

※テストデータ

| | サイズ | エンコード | 内容 |
|---|---|---|---|
| ファイル1 | 100 B | JIS | 日本語テキストファイル |
| ファイル2 | 100 B | SHIFT-JIS | HTMLファイル |
| ファイル3 | 100 B | EUC | C言語ソースファイル |
| ファイル4 | 100 B | UNICODE | 3バイト文字を含むファイル |
| ファイル5 | 1 KB | SHIFT-JIS | 日本語テキストファイル |
| ファイル6 | 1 KB | JIS | HTMLファイル |
| ファイル7 | 1 KB | UNICODE | C言語ソースファイル |
| ファイル8 | 1 KB | EUC | 3バイト文字を含むファイル |
| ファイル9 | 10 KB | EUC | 日本語テキストファイル |
| ファイル10 | 10 KB | UNICODE | HTMLファイル |
| ファイル11 | 10 KB | JIS | C言語ソースファイル |
| ファイル12 | 10 KB | SHIFT-JIS | 3バイト文字を含むファイル |
| ファイル13 | 100 B | UNICODE | 日本語テキストファイル |
| ファイル14 | 100 B | EUC | HTMLファイル |
| ファイル15 | 100 B | SHIFT-JIS | C言語ソースファイル |
| ファイル16 | 100 B | JIS | 3バイト文字を含むファイル |

# スープカレー方式の利点

- システムテストの切り口が明確になる
- 機能網羅と顧客観点の網羅を同時に行なえる
- 機能観点×顧客観点のマトリクスで、発想が広がる
- メンバーに観点を明示でき、共有できる
- FV表やFL表などに繋げやすい

TEF-道

How do you like software test?

# 課題点

- システムテストの範囲の全てを網羅してはいない
- 手間がかかる

  ⇒設計時に作成しておくと良いかもしれない
- スープカレー表のレイアウトは、まだ議論の余地あり

TEF-道

How do you like software test?

# ありがとうございました！

ごちそう
さまでした

TEF-道
How do you like software test?