JaSST ’19 Tokyo

FUJITSU
shaping tomorrow with you

テスト自動化を10年続けて分かったこと
- 開発者が語るINTARFRMテストツール -

富士通株式会社

自己紹介

頭島　俊樹（かしらじま　としき）

システム開発のための
技術支援やツール開発

所属

富士通株式会社
サービステクノロジー本部
システムインテグレーション技術統括部

主な仕事

富士通製アプリケーションフレームワーク
INTARFRM（インターファーム）
の開発

本日の内容

FUJITSU

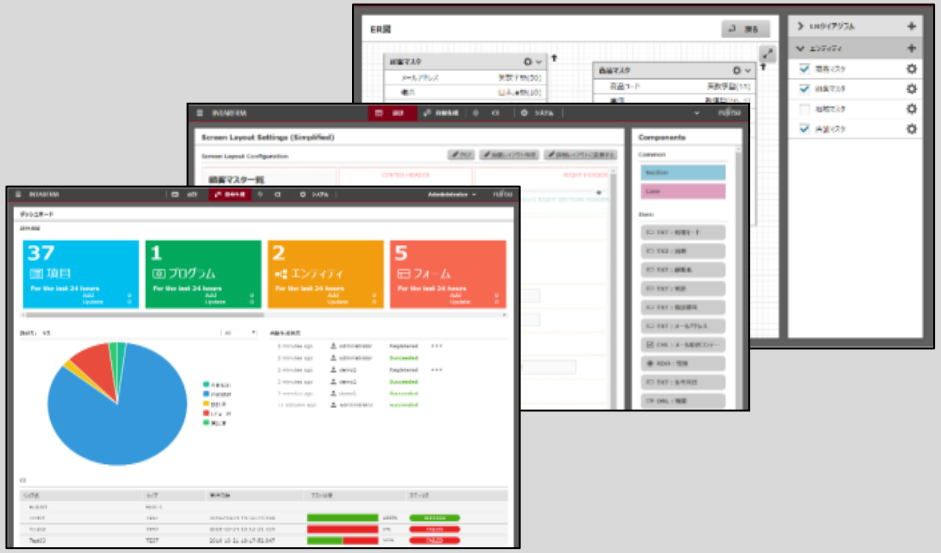
- テスト自動化を始めた背景
- 過去の取り組み紹介
- 現在の取り組み紹介
- まとめ

テスト自動化を始めた背景

私が担当している製品のご紹介

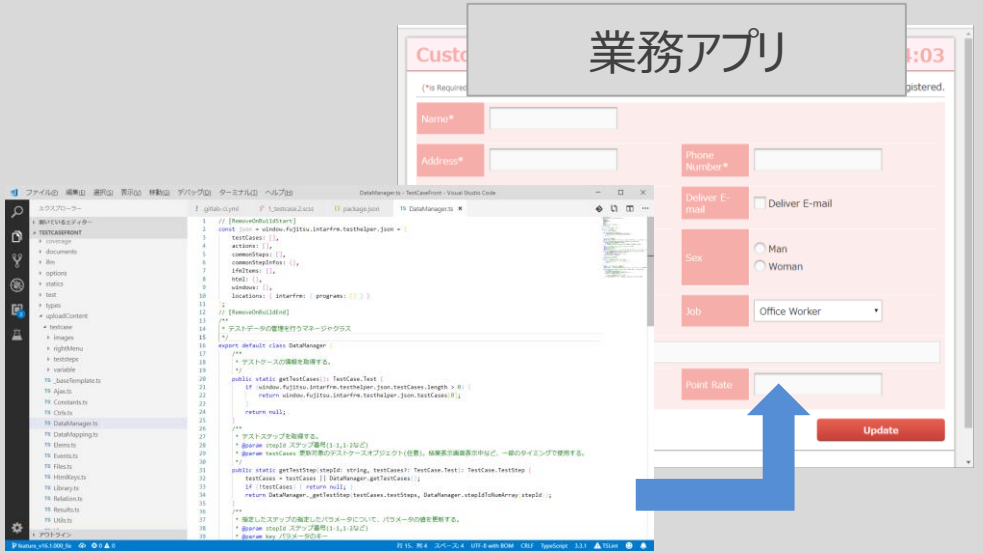
アプリケーション開発を支えるフレームワーク「INTARFRM（インターファーム）」

「設計」支援機能



特徴
- 専用の設計ツール
- 設計情報の管理・分析
- 設計内容を元にしたソースコードの生成

「開発」支援機能



特徴
- 値のフォーマット処理、データベース処理など「よくある実装」を部品として提供
- Java/C# など様々な言語・プラットフォームに対応

私が担当している製品のご紹介

FUJITSU

アプリケーション開発を支えるフレームワーク「INTARFR

「設計」支援機能

主な機能
・アプリケーション設計ツールによる設計情報の登録
・設計内容を元にしたソースコードの生成

私が担当している製品のご紹介

FUJITSU

アプリケーション開発を支えるフレームワーク「INTARFR

「設計」支援機能

主な機能
・アプリケーション設計ツールによる設計情報の登録
・設計内容を元にしたソースコードの生成

テスト自動化が必須

～2009年
キャプチャ & リプレイ編

キャプチャ & リプレイ

キャプチャ
(記録)

リプレイ
(再生)

記録内容

1. 「会社登録画面」を開く
2. 「新規ボタン」をクリック
3. 「会社名」に「富士通」を入力する

キャプチャ & リプレイ

キャプチャ
（記録）

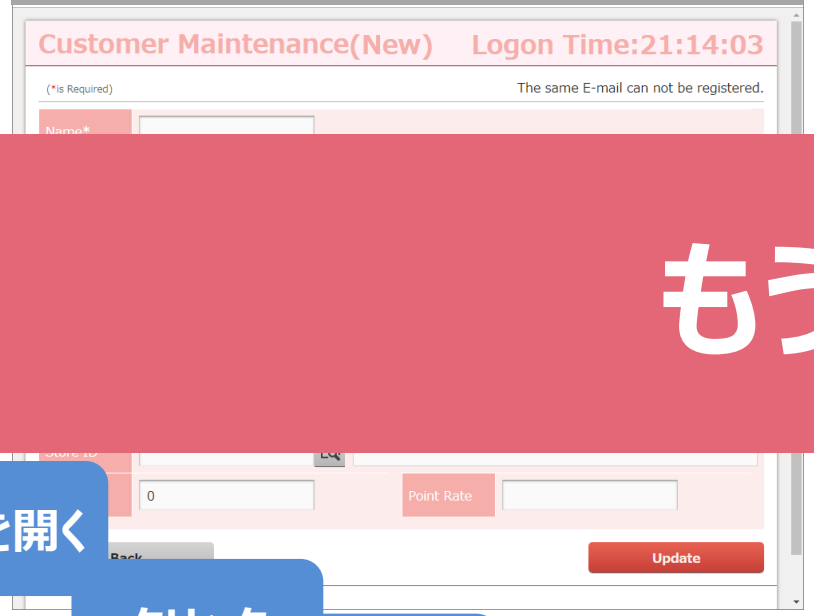
記録内容

1. 「会社登録画面」を開く
2. 「変更ボタン」をクリック
3. 「会社名」に「富士通」を入力する

キャプチャ & リプレイ

FUJITSU

ブラウザ



キャプチャ
（記録）

記録内容

1 「会社登録画面」を開く

3 「会社名」に
「富士通」を入力する

もう一回やり直し

実例：TestPartnerで記録された内容

FUJITSU

テストコード（VBA）

```
IEWindow("Application = IEXPLORE.EXE")
HTMLBrowser("Caption='*'").Open "http://localhost/TestApp/TestPage20"
HTMLComboBox("Name=StoreName").Select "東京都蒲田店"
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLButton("Name=Register").Click
HTMLComboBox("Name=StoreName").Select "東京都蒲田店"
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLInput("Name=StoreTel").Text "0123-123-456"
HTMLButton("Name=Register").Click
HTMLComboBox("Name=StoreName").Select "東京都蒲田店"
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLInput("Name=StoreTel").Text "0123-123-456"
HTMLInput("Name=OwnerName").Text "富士通太郎"
HTMLButton("Name=Register").Click
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLButton("Name=Register").Click
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLInput("Name=StoreTel").Text "0123-123-456"
HTMLButton("Name=Register").Click
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLInput("Name=StoreTel").Text "0123-123-456"
HTMLInput("Name=OwnerName").Text "富士通太郎"
HTMLButton("Name=Register").Click
HTMLButton("Name=Register").Click
```

※スクリプトは一部抜粋

IEにURLを渡して画面を開いている

値を入力してボタンをクリックして・・・

どこまでが一連の流れ？そもそも正常系？異常系

実例：TestPartnerで記録された内容

FUJITSU

テストコード（VBA）

```
IEWindow("Application = IEXPLORE.EXE")
HTMLBrowser("Caption='*'").Open "http://localhost/TestApp/TestPage20"
HTMLComboBox("Name=StoreName").Select "東京都蒲田店"
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLButton("Name=Register").Click
```

IEにURLを渡して
画面を開いている

操作の記録 ≠ テストの記録

```
HTMLButton("Name=Register").Click
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLButton("Name=Register").Click
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLInput("Name=StoreTel").Text "0123-123-456"
HTMLButton("Name=Register").Click
HTMLInput("Name=StoreAddress").Text "東京都大田区大森西X-X-X"
HTMLInput("Name=StoreTel").Text "0123-123-456"
HTMLInput("Name=OwnerName").Text "富士通太郎"
HTMLButton("Name=Register").Click
HTMLButton("Name=Register").Click
```

どこまでが一連の流れ？
そもそも正常系？異常系

※スクリプトは一部抜粋

キャプチャ ＆ リプレイ

良かった点

■導入が簡単

- 自動化のための作業が少ない
- 覚えることも少ない

しかし・・・

■内容を後で読みづらい

■追加・修正＝再記録で手間

■「操作」以外は記録不可

2010年 ～ 2013年
テストフレームワークで実装編

テストフレームワークを用いた実装

FUJITSU

実現したいテスト

1. 「会社登録画面」を開く
2. 「新規ボタン」をクリックする
3. 「会社名」に「富士通」を入力する

開発環境

1. `Window= GetWindow();`
   `Window.Open(“会社登録画面”);`
2. `New= GetItem(“新規ボタン”)`
   `New.Click();`
3. `Com = GetItem(“会社名”);`
   `Com.Input(“富士通”);`

テストフレームワーク

代表例：Selenium、Puppeteer　など

テストフレームワークを用いた実装

FUJITSU

実現したいテスト

1. 「会社登録画面」を開く
2. 「新規ボタン」をクリックする
3. 「会社名」に「富士通」を入力する

開発環境

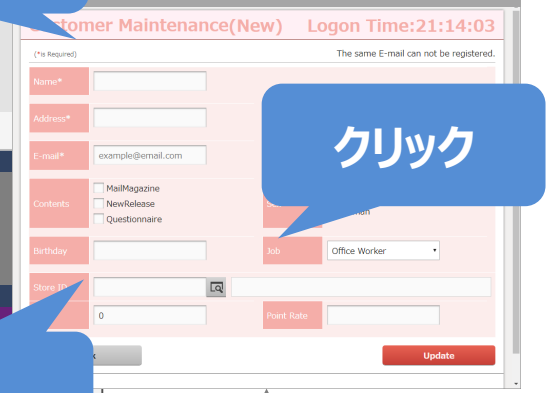
```
1  Window= GetWindow();
   Window.Open("会社登録画面");

2  New= GetItem("新規ボタン")
   New.Click();

3  Com = GetItem(
   Com.Input("富士通");
```

テストフレームワーク

代表例：Selenium、Puppet

画面を開く

ブラウザ



クリック

入力

実例：TestPartnerを使ったテストコード（1）

FUJITSU

コメントで内容整理

テストコード（VBA）

```
' ★ テストケース①：店舗名を選んで登録するテスト
' 実測値
Dim Actual As String
' 期待値
Dim Expected As String

' (1)IEを起動する
Call RunInternetExplorer("http://localhost/TestApp/TestPage20")

' (2)店舗を選択する
HTMLComboBox("Name=StoreName").Select "東京都蒲田店"

' (3)登録ボタンをクリックする
HTMLButton("Name=Register").Click

' (4)登録に失敗するメッセージが表示されていることを確認する
Actual = HTMLInput("Name=Message")
Expected = "登録に失敗しました"
CheckResult  Actual, Expected
```

処理の共通化

値の検証

実例：TestPartnerを使ったテストコード（2）

読むのが大変

テストコード（VBA）

```
‘ ★ テストケース②：店舗名のバリエーションテスト
Dim Actual As String
Dim Expected As String
Call RunInternetExplorer("http://localhost/TestApp/TestPage20")
HTMLComboBox("Name=StoreName").Select "東京都蒲田店"
HTMLButton("Name=Register").Click
Actual = HTMLInput("Name=Message")
Expected = "登録に失敗しました"
CheckResult  Actual, Expected
HTMLComboBox( “Name=StoreName” ).Select "東京都品川店"
HTMLButton("Name=Register").Click
Actual = HTMLInput("Name=Message")
Expected = "登録に失敗しました"
CheckResult  Actual, Expected
HTMLComboBox( “Name=StoreName” ).Select "神奈川県川崎店"
HTMLButton("Name=Register").Click
Actual = HTMLInput("Name=Message")
Expected = "登録に失敗しました"
CheckResult  Actual, Expected
```

類似処理のコピペ

実例：TestPartnerを使ったテストコード（2）

読むのが大変

テストコード（VBA）

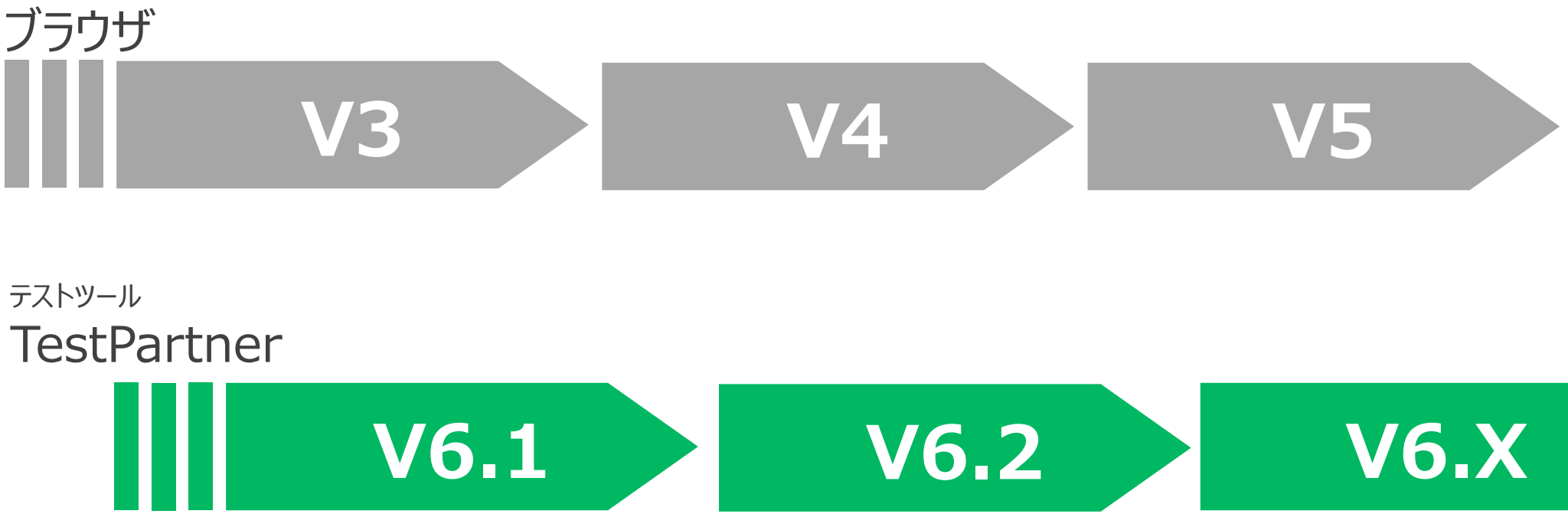
```
‘ ★ テストケース②：店舗名のバリエーションテスト
Dim Actual As String
Dim Expected As String
Call RunInternetExplorer("http://localhost/TestApp/TestPage20")
HTMLComboBox("Name=StoreName").Select "東京都蒲田店"
Actual = HTMLInput("Name=Message")
Expected = "登録に失敗しました"
CheckResult  Actual, Expected
HTMLComboBox( “Name=StoreName” ).Select "神奈川県川崎店"
HTMLButton("Name=Register").Click
Actual = HTMLInput("Name=Message")
Expected = "登録に失敗しました"
CheckResult  Actual, Expected
```

メンテが楽か は 実装者次第

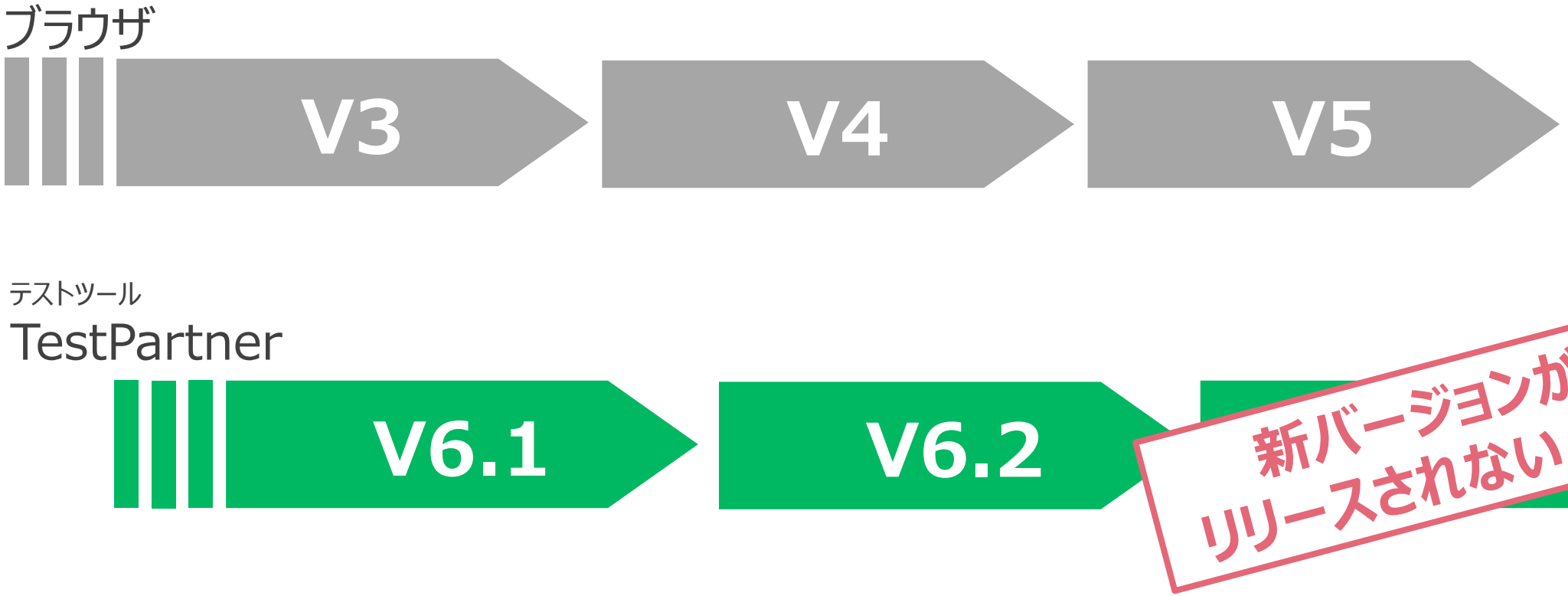
類似処理のコピペ

数年後・・・

ブラウザのバージョンアップ



ブラウザのバージョンアップ



ブラウザのバージョンアップ

実例：テストフレームワークの移行

テストコード（VBA）

```
' 期待値
Dim Expected As String
' テスト対象項目ID
Dim ItemId As String


'IE起動
Call RunInternetExplorer(ProgramId, vbNormalFocus)
'ブラウザにアタッチ
IEWindow("Application = IEXPLORE.EXE Caption='" & ProgramNm & " - *'
HTMLBrowser("Caption=" & ProgramNm).Attach

    'ドロップダウンリストから選択する
    HTMLComboBox("Name=Sikucode").Select "12101:千葉県 -- 千葉市中央
    HTMLComboBox("Name=Sikucode").Select "13105:東京都 -- 文京区"

    '★正しく選択されていることを確認する
    TestCaseName = "ドロップダウンリストが正しく選択されていることを
    ItemId = "Name=Sikucode"
    Actual = HTMLComboBox("Name=Sikucode").Text
    Expected = "13105:東京都 -- 文京区"
    '結果をチェックする
    CheckResult FormId, TestCaseName, ItemId, Actual, Expected

    '次へボタンを押下する
    HTMLButton("Name=Btnenter").SetFocus
    HTMLButton("Name=Btnenter").Click

    '★ドロップダウンリストの確認
    TestCaseName = "ドロップダウンリストが正しく選択されていることを
    ItemId = "ID='' Index=3"
    Actual = Replace(HTMLTD("ID='' Index=3").Text, " ", "")
    Expected = "----選択して下さい----白山（２～５丁目）小石川春日後
    '結果をチェックする
    CheckResult FormId, TestCaseName, ItemId, Actual, Expected

    '★明細件数の確認
    TestCaseName = "明細件数を確認する"
    ItemId = "ID='' Index=4"
    Actual = Replace(Replace(HTMLTD("ID='' Index=4").Text, " ", ""),
    Expected = "6件中1-5件を表示しています。"
    '結果をチェックする
    CheckResult FormId, TestCaseName, ItemId, Actual, Expected

    '★検索結果の確認
    TestCaseName = "検索結果を確認する"
```

FWの差

言語の差

テストコード（Java）

```
    // ---------------------------------------------------------------------
    // [チェックON]桁数チェックによってエラーメッセージが表示されることを確認します。
    // ---------------------------------------------------------------------
//    // ブラウザを閉じます。
//    control.closeAllBrowsers();
    // データ設定
    setDbTable(IN008_001);
//    // 画面を起動します。
//    control.startApp(testInfo.getUrl()[INDEX_TESTFORM]);
//    browser = control.getNewBrowserObject();
    // データの読み込みを行います。
    control.loadBrowser(browser, testInfo.getUrl()[INDEX_TESTFORM]);
    // 画面のロードを待機します。
    control.waitForBrowserLoading(browser);
    // Formオブジェクトを取得します。
    form = control.getForm(browser, formId);

    // テキストボックスにテストデータを設定します。
    setItemValue(control, browser, form, IN008_002);
    // 入力チェック
    control.buttonClick(form, formVo.getItemViewId(Mode1001Form.BTNCMNCHECK));
    // 画面のロードを待機します。
    control.waitForBrowserLoading(browser);
    // Formオブジェクトを取得します。
    form = control.getForm(browser, formId);
    //エラーメッセージ
    serverErrorCheck(control, testInfo, EX008_001, formId);
    //項目のスタイル
    attributeCheck(control, browser, testInfo, EX008_002, formId, Browser.ATTRIBUTE_

    // ---------------------------------------------------------------------
    // [チェックOFF]桁数チェックが実行されずエラーメッセージが表示されないことを確認し
```

実例：テストフレームワークの移行

FUJITSU

テストコード（VBA）

```
' 期待値
Dim Expected As String
' テスト対象項目ID
Dim ItemId As String


'IE起動
Call RunInternetExplorer(ProgramId, vbNormalFocus)
'ブラウザにアタッチ
IEWindow("Application = IEXPLORE.EXE Caption='" & ProgramNm & " - *'
HTMLBrowser("Caption=" & ProgramNm).Attach

    'ドロップダウンリストから選択する
    HTMLComboBox("Name=Sikucode").Select "12101:千葉県 -- 千葉市中央
    ItemId = "ID='' Index=3"
    Actual = Replace(HTMLTD("ID='' Index=3").Text, " ", "")
    Expected = "----選択して下さい----白山（２～５丁目）小石川春日後
    '結果をチェックする
    CheckResult FormId, TestCaseName, ItemId, Actual, Expected

    '★明細件数の確認
    TestCaseName = "明細件数を確認する"
    ItemId = "ID='' Index=4"
    Actual = Replace(Replace(HTMLTD("ID='' Index=4").Text, " ", ""),
    Expected = "6件中1-5件を表示しています。"
    '結果をチェックする
    CheckResult FormId, TestCaseName, ItemId, Actual, Expected

    '★検索結果の確認
    TestCaseName = "検索結果を確認する"
```

テストコード（Java）

```
    // ---------------------------------------------------------------------
    // [チェックON]桁数チェックによってエラーメッセージが表示されることを確認します。
    // ---------------------------------------------------------------------
//  // ブラウザを閉じます。
//  control.closeAllBrowsers();
    // データ設定
    setDbTable(IN008_001);
//  // 画面を起動します。
//  control.startApp(testInfo.getUrl()[INDEX_TESTFORM]);
//  browser = control.getNewBrowserObject();

    // 画面のロードを待機します。
    control.waitForBrowserLoading(browser);
    // Formオブジェクトを取得します。
    form = control.getForm(browser, formId);
    //エラーメッセージ
    serverErrorCheck(control, testInfo, EX008_001, formId);
    //項目のスタイル
    attributeCheck(control, browser, testInfo, EX008_002, formId, Browser.ATTRIBUTE_

    // ---------------------------------------------------------------------
    // [チェックOFF]桁数チェックが実行されずエラーメッセージが表示されないことを確認し
```

テストに力を入れてた分
作業も膨大・・・

テストフレームワークで実装

キャプチャ & リプレイに比べて

- ■柔軟な記述が可能
  - •コメント / 共通化 / 条件分岐 / ループ ・・・
  - •外部ライブラリとの連携（Excel, DB, など）

しかし・・・

- ■テストコードの品質は作成者に依存する
- ■テストツール変更時の移行コストが高い

2014年 ～ 2015年
キーワード駆動テスト編

プログラミングの流れ

画面を開くには
「GetWindow」でウィンドウを取得して
パラメータとして「会社登録画面」を渡して
ウェイトして・・・

実現したいテスト		自動化プログラム	
1	「会社登録画面」を開く	1	Window= GetWindow(); Window.Open(“会社登録画面”); Wait(10);
2	「新規ボタン」をクリックする	2	New= GetItem(”新規ボタン”) New.Click();
3	「会社名」に「富士通」を入力する	3	COM = GetItem(“会社名”); COM.Input(“富士通”);

プログラミングの流れ

画面を開くには
「GetWindow」でウィンドウを取得して
パラメータとして「会社登録画面」を渡して
ウェイトして・・・

機械的に処理できそう？

	を開く	Wait(10);
2	「新規ボタン」をクリックする	New= GetItem(”新規ボタン”) New.Click();
3	「会社名」に「富士通」を入力する	COM = GetItem(“会社名”); COM.Input(“富士通”);

キーワード化

FUJITSU

実現したいテスト	キーワード形式
1 「会社登録画面」を開く	1 キーワード：画面を開く パラメータ：会社登録画面
2 「新規ボタン」をクリックする	2 キーワード：クリックする パラメータ：新規ボタン
3 「会社名」に「富士通」を入力する	3 キーワード：値を入力する パラメータ：会社名、富士通

操作内容
(キーワード)

操作が指定する値
(パラメータ)

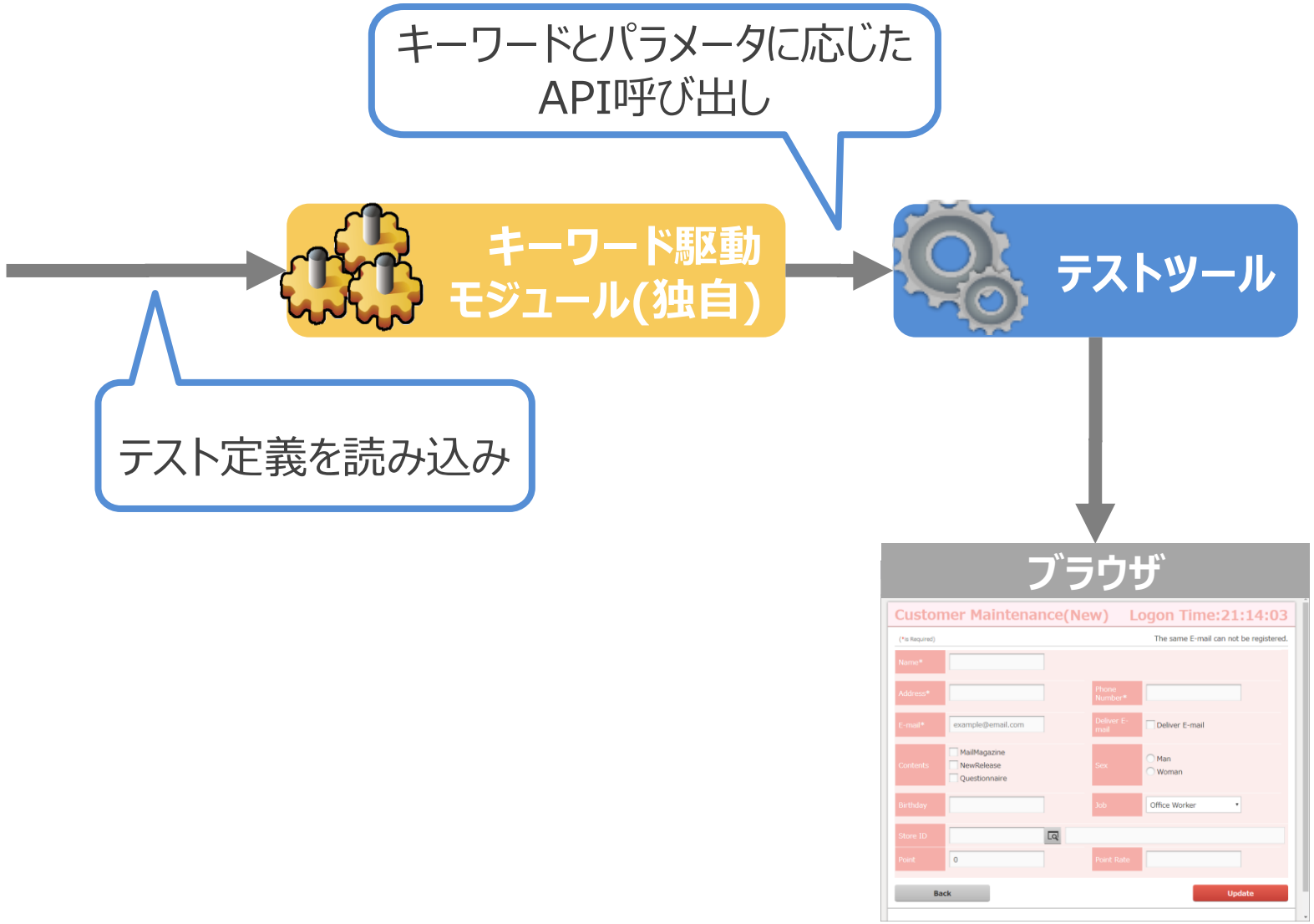
**型化することで
機械的に処理できる**

キーワード駆動

FUJITSU

テスト定義

1	画面を開く 会社登録画面
2	クリックする 新規ボタン
3	値を入力する 会社名、富士通
4	値を入力する 都道府県、東京
5	クリックする 登録ボタン
6	値を確認する メッセージ、登録OK



実例：Excelを用いたキーワード駆動テスト

FUJITSU

操作内容(=キーワード)

パラメータ

テスト定義ファイル（Excel）

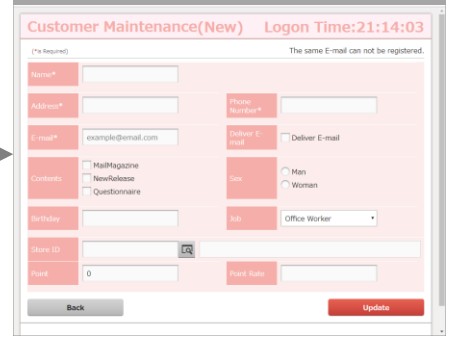
	A	B	C	D	E	F	G
1	No	テスト内容	操作内容	URL	項目ID	値	
2	1	画面の起動	画面起動	http://localhost/testapp/txtnb001			
3	2		値入力		TEXT_FORMAT_01	-a1234	
4	3		クリック		BUTTON		
5	4	TEXT_FORMAT_01の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。	
6	5		値入力		TEXT_FORMAT_02	-a1234	
7	6		クリック		BUTTON		
8	7	TEXT_FORMAT_02の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。	
9	8		値入力		TEXT_FORMAT_03	-a1234	
10	9		クリック		BUTTON		
11	10	TEXT_FORMAT_03の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。	
12	11		画面終了				

上から
順番に定義

キーワード駆動
モジュール
for RFT

RFT

ブラウザ



コーディングが無くなり
記述が統一される

テストツールを意識せず
テストの定義に集中

実例：Excelを用いたキーワード駆動テスト

FUJITSU

操作内容(=キーワード)

パラメータ

テストファイル（Excel）

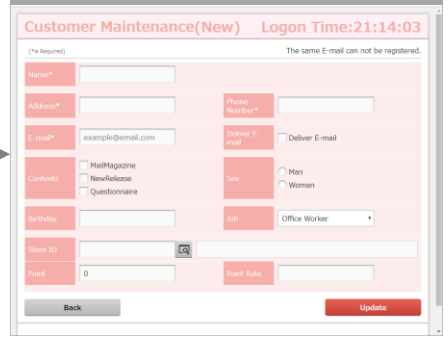
	A	B	C	D	E	F	G
1	No	テスト内容	操作内容	URL	項目ID	値	
2	1	画面の起動	画面起動	http://localhost/testapp/txtnb001			
3	2		値入力		TEXT_FORMAT_01	-a1234	
4	3		クリック		BUTTON		
5	4	TEXT_FORMAT_01の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。	
6	5		値入力		TEXT_FORMAT_02	-a1234	
7	6		クリック		BUTTON		
8	7	TEXT_FORMAT_02の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。	
9	8		値入力		TEXT_FORMAT_03	-a1234	
10	9		クリック		BUTTON		
11	10	TEXT_FORMAT_03の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。	
12	11		画面終了				

上から
順番に定義

キーワード駆動
モジュール
for RFT

新バージョンが
リリースされない！？

ブラウザ



コーディングが無くなり
記述が統一される

テストツールを意識せず
テストの定義に集中

実例：Excelを用いたキーワード駆動テスト

FUJITSU

操作内容(=キーワード)

パラメータ

テスト定義ファイル（Excel）

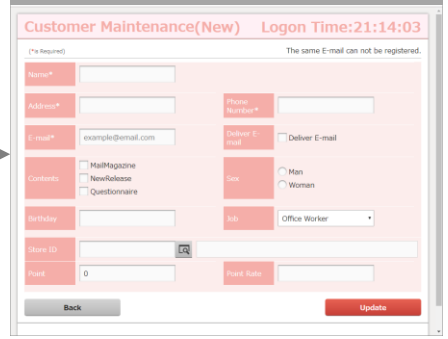
上から
順番に定義

	A	B	C	D	E	F	G
1	No	テスト内容	操作内容	URL	項目ID	値	
2	1	画面の起動	画面起動	http://localhost/testapp/txtnb001			
3	2		値入力		TEXT_FORMAT_01	-a1234	
4	3		クリック		BUTTON		
5	4	TEXT_FORMAT_01の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。	
6	5		値入力		TEXT_FORMAT_02	-a1234	
7	6		クリック		BUTTON		
8	7	TEXT_FORMAT_02の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。	
9	8		値入力		TEXT_FORMAT_03	-a1234	
10	9		クリック		BUTTON		
11	10	TEXT_FORMAT_03の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。	
12	11		画面終了				

キーワード駆動
モジュール
for Selenium

Selenium

ブラウザ



コーディングが無くなり
記述が統一される

テストツールを意識せず
テストの定義に集中

テストツールの
変更も容易

1年後・・・

実例：Excelを用いたキーワード駆動テスト②

様々な機能を追加

No	テスト内容	操作内容	URL	項目ID	値
1	画面の起動	画面起動	http://localhost/testapp/txtnb001		
2		値入力		TEXT_FORMAT_01	-a1234
3		クリック		BUTTON	
4	TEXT_FORMAT_01の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。
5	TEXT_FORMAT_01のスタイルのチェック	スタイルの確認		TEXT_FORMAT_01	

実例：Excelを用いたキーワード駆動テスト②

様々な機能を追加

■「スタイルのチェック機能」を追加

	A	B	C	D	E	F	G	H
1	No	テスト内容	操作内容	URL	項目ID	値	チェック対象スタイル	
2	1	画面の起動	画面起動	http://localhost/testapp/txtnb001				
3	2		値入力		TEXT_FORMAT_01	-a1234		
4	3		クリック		BUTTON			
5	4	TEXT_FORMAT_01の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。		
6	5	TEXT_FORMAT_01のスタイルのチェック	スタイルの確認		TEXT_FORMAT_01		color:red	
7								
8								

実例：Excelを用いたキーワード駆動テスト②

様々な機能を追加

- 「スタイルのチェック機能」を追加
- 「テーブルデータ削除機能」を追加

	A	B	C	D	E	F	G	H
1	No	テスト内容	操作内容	URL	項目ID	値	チェック対象スタイル	テーブル名
2	1	画面の起動	画面起動	http://localhost/testapp/txtnb001				
3	2		値入力		TEXT_FORMAT_01	-a1234		
4	3		クリック		BUTTON			
5	4	TEXT_FORMAT_01の異常値入力時のチェック	値の確認		MESSAGE	正しくない値が入力されています。		
6	5	TEXT_FORMAT_01のスタイルのチェック	スタイルの確認		TEXT_FORMAT_01		color:red	
7	6	データのクリア	テーブルデータの削除					M_STORE
8								

実例：Excelを用いたキーワード駆動テスト②

様々な機能を追加

- 「スタイルのチェック機能」を追加
- 「テーブルデータ削除機能」を追加
- その他色々追加・・・

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	no	testItemId	testItemNote	actionId	actionName	checkMessage	windowId	pgId	formId	groupId	listId	codeId	itemId	rowIndex	columnIndex	value	selectState	conditionValue	dxFormat	styleName	propertyName	exists	tableId	columnId	dataFile	note
2	1	PRE		TableDataDelete	テーブルデータ全削除																		T_TXTNB001			既存データの削除
3	2	PRE		TableDataDelete	テーブルデータ全削除																		M1_TXTNB001			既存データの削除
4	3	PRE		TableDataDelete	テーブルデータ全削除																		M2_TXTNB001			既存データの削除
5	4		[サーバー側]異常値のデータロード後フォ	TableDataInsert	テーブルデータ複数レコード登録																		T_TXTNB001		C:¥Rft¥1.4¥AutoTest	データロードテスト用のデータを挿入
6	5	1		TableDataInsert	テーブルデータ複数レコード登録																		M1_TXTNB001		C:¥Rft¥1.4¥AutoTest	データロードテスト用のデータを挿入
7	6	1		TableDataInsert	テーブルデータ複数レコード登録																		M2_TXTNB001		C:¥Rft¥1.4¥AutoTest	データロードテスト用のデータを挿入
8	7	1		WindowOpen	画面起動		Windows1	TXTNB001	TXTNB001																	画面起動
9	8	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB10			-1234	FALSE									画面のデータ確認(カード部)
10	9	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB11			-1,234.5	FALSE									画面のデータ確認(カード部)
11	10	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB12			-1,234.5	FALSE									画面のデータ確認(カード部)
12	11	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB14			-1234.5	FALSE									画面のデータ確認(カード部)
13	12	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB15			-1,234.5	FALSE									画面のデータ確認(カード部)
14	13	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB18			-1234.50	FALSE									画面のデータ確認(カード部)
15	14	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB19			-1,234.50	FALSE									画面のデータ確認(カード部)
16	15	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB21			-1234.50	FALSE									画面のデータ確認(カード部)
17	16	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB22			-1,234.50	FALSE									画面のデータ確認(カード部)
18	17	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB31			-¥1,234.5	FALSE									画面のデータ確認(カード部)
19	18	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB32			-$1,234.5	FALSE									画面のデータ確認(カード部)
20	19	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB34			-¥1,234.5	FALSE									画面のデータ確認(カード部)
21	20	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB35			-$1,234.5	FALSE									画面のデータ確認(カード部)
22	21	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB37			-¥1,234.50	FALSE									画面のデータ確認(カード部)
23	22	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(カード部)			TXTNB001				TXTNB38			-$1,234.50	FALSE									画面のデータ確認(カード部)
24	23	1	[サーバー側]異常値のデータロード後フォ	TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB10M1	1		-1234	FALSE									画面のデータ確認(明細部M1)
25	24	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB11M1	1		-1234.5	FALSE									画面のデータ確認(明細部M1)
26	25	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB12M1	1		-1,234.5	FALSE									画面のデータ確認(明細部M1)
27	26	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB14M1	1		-1234.5	FALSE									画面のデータ確認(明細部M1)
28	27	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB15M1	1		-1,234.5	FALSE									画面のデータ確認(明細部M1)
29	28	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB18M1	1		-1234.50	FALSE									画面のデータ確認(明細部M1)
30	29	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB19M1	1		-1,234.50	FALSE									画面のデータ確認(明細部M1)
31	30	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB21M1	1		-1234.50	FALSE									画面のデータ確認(明細部M1)
32	31	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB22M1	1		-1,234.50	FALSE									画面のデータ確認(明細部M1)
33	32	1		TextFormatCheck	入力値のフォーマット確認	【サーバー側】異常値データをロードした場合のフォーマットを確認します。(明細部M1)			TXTNB001				TXTNB31M1	1		-¥1,234.5	FALSE									画面のデータ確認(明細部M1)

キーワード駆動

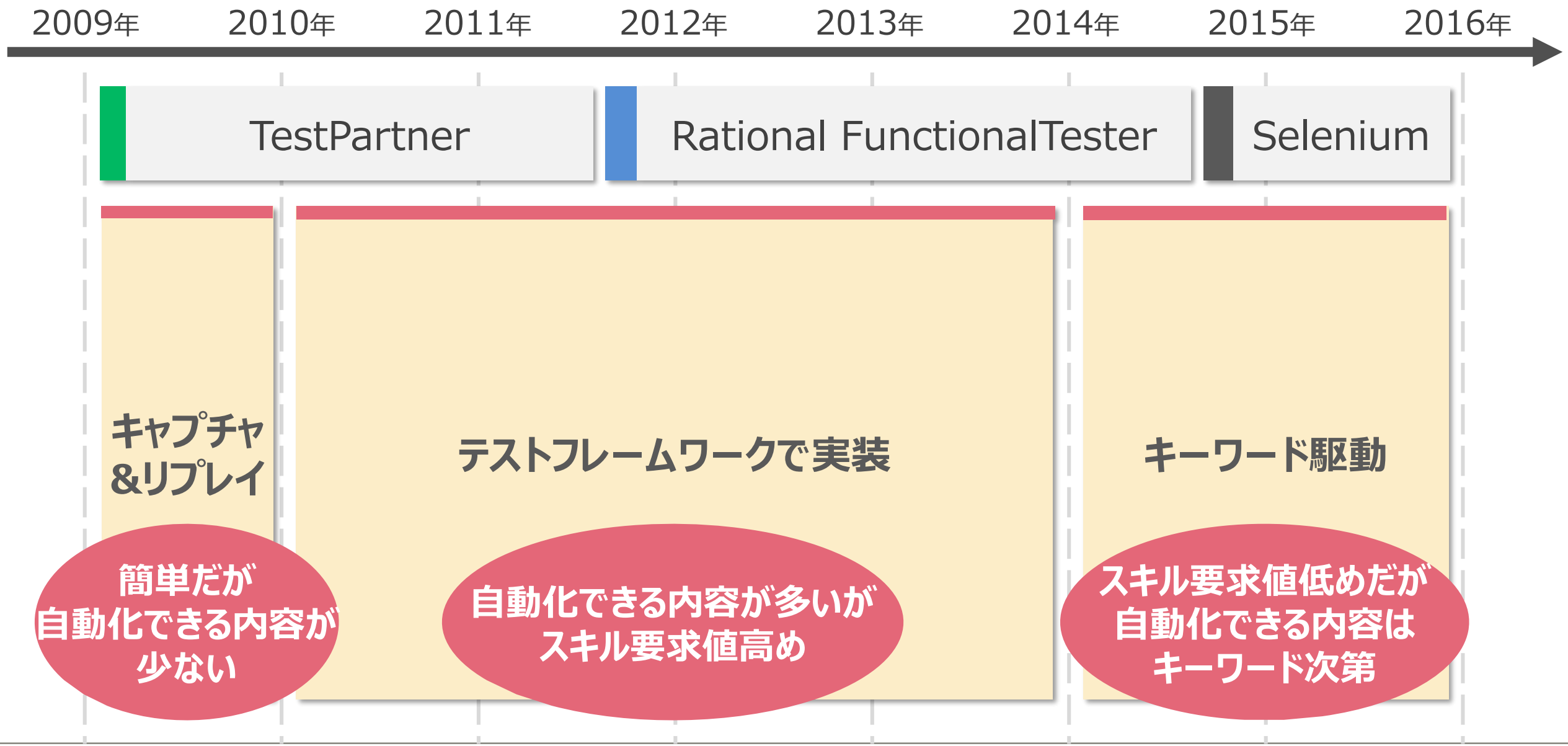
テストフレームワークでの実装に比べて

- ■プログラミングが不要
  - ●属人性低下、可読性の安定
- ■テストツールの移行が可能

しかし・・・

- ■キーワードの組み合わせへの落とし込みが難しい
  - ●独自のルールの把握
  - ●キーワードのバリエーション　＝　実現範囲

テスト自動化の振り返り



簡単に作れて
メンテが楽で
長く使える
ツールが欲しい！！

作りました

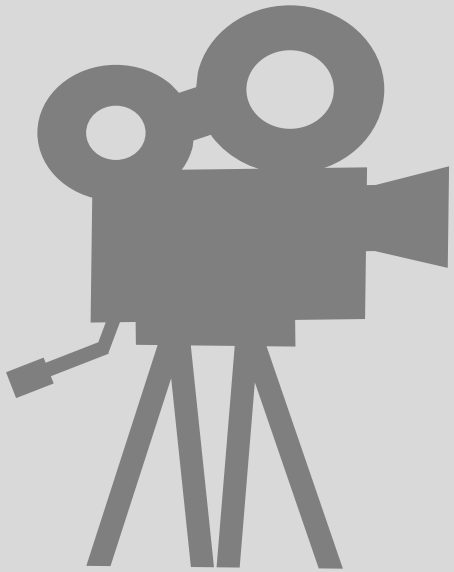
FUJITSU

2016年 ～ 2019年
INTARFRMテストレコーディングツール編

コンセプト

FUJITSU

簡単に作れる

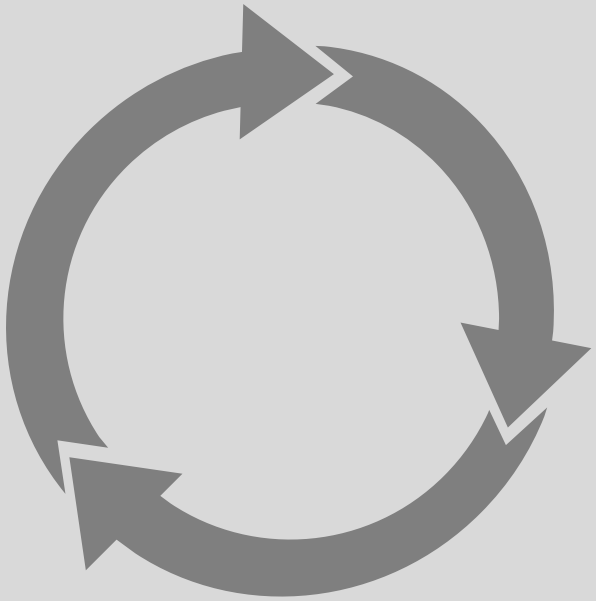


・操作の記録
・操作以外も登録できるGUI

高いメンテナンス性

・可読性の高い画面
・操作の編集機能

長く使える



テスト設計（JSON）
テストツール（Selenium）
の疎結合化

基本機能

FUJITSU

記録機能

編集機能

テスト実行機能

レポート出力機能

記録機能

上から
順番に記録

テストケースエディタ

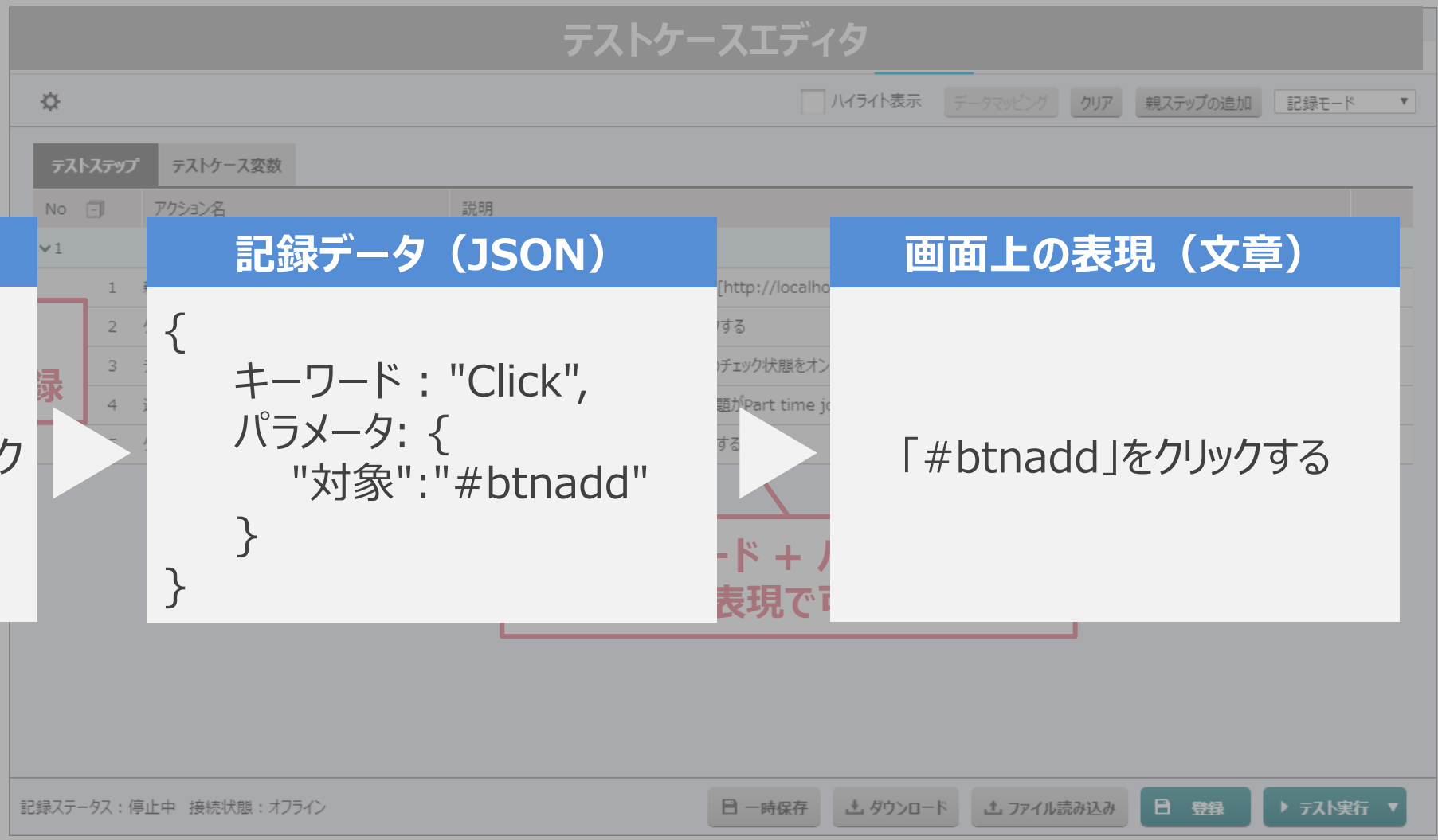
ハイライト表示　データマッピング　クリア　親ステップの追加　記録モード

テストステップ　テストケース変数

No	アクション名	説明	
1		Recording test step.	
1	新規ウインドウを開く	http://localhost/Demo/tyu010we [http://localhost/Demo/tyu010we] を新規ウィンドウ(ウィンドウ名 、ウィンドウスタイル)で開く	
2	クリック	セレクター:[tyu01002-btnadd]をクリックする	
3	チェック状態の変更	セレクター:[tyu01001-deliveremail]のチェック状態をオンにする	
4	選択肢項目の選択(ラベル指定)	セレクター:[tyu01001-job]について、標題がPart time jobと一致する項目を選択状態にする	
5	クリック	セレクター:[tyu01001-btndef]をクリックする	

記録ステータス：停止中　接続状態：オフライン　一時保存　ダウンロード　ファイル読み込み　登録　テスト実行

記録機能



ユーザーの操作	記録データ（JSON）	画面上の表現（文章）
登録ボタン(＃btnadd)のクリック	{ キーワード："Click", パラメータ: { "対象":"#btnadd" } }	「＃btnadd」をクリックする

記録機能

上から
順番に記録

テストケースエディタ

ハイライト表示　データマッピング　クリア　親ステップの追加　記録モード

テストステップ　テストケース変数

No	アクション名	説明	
1		Recording test step.	
1	新規ウインドウを開く	http://localhost/Demo/tyu010we [http://localhost/Demo/tyu010we] を新規ウィンドウ(ウィンドウ名 、ウィンドウスタイル)で開く	
2	クリック	セレクター:[tyu01002-btnadd]をクリックする	
3	チェック状態の変更	セレクター:[tyu01001-deliveremail]のチェック状態をオンにする	
4	選択肢項目の選択(ラベル指定)	セレクター:[tyu01001-job]について、標題がPart time jobと一致する項目を選択状態にする	
5	クリック	セレクター:[tyu01001-btndef]をクリックする	

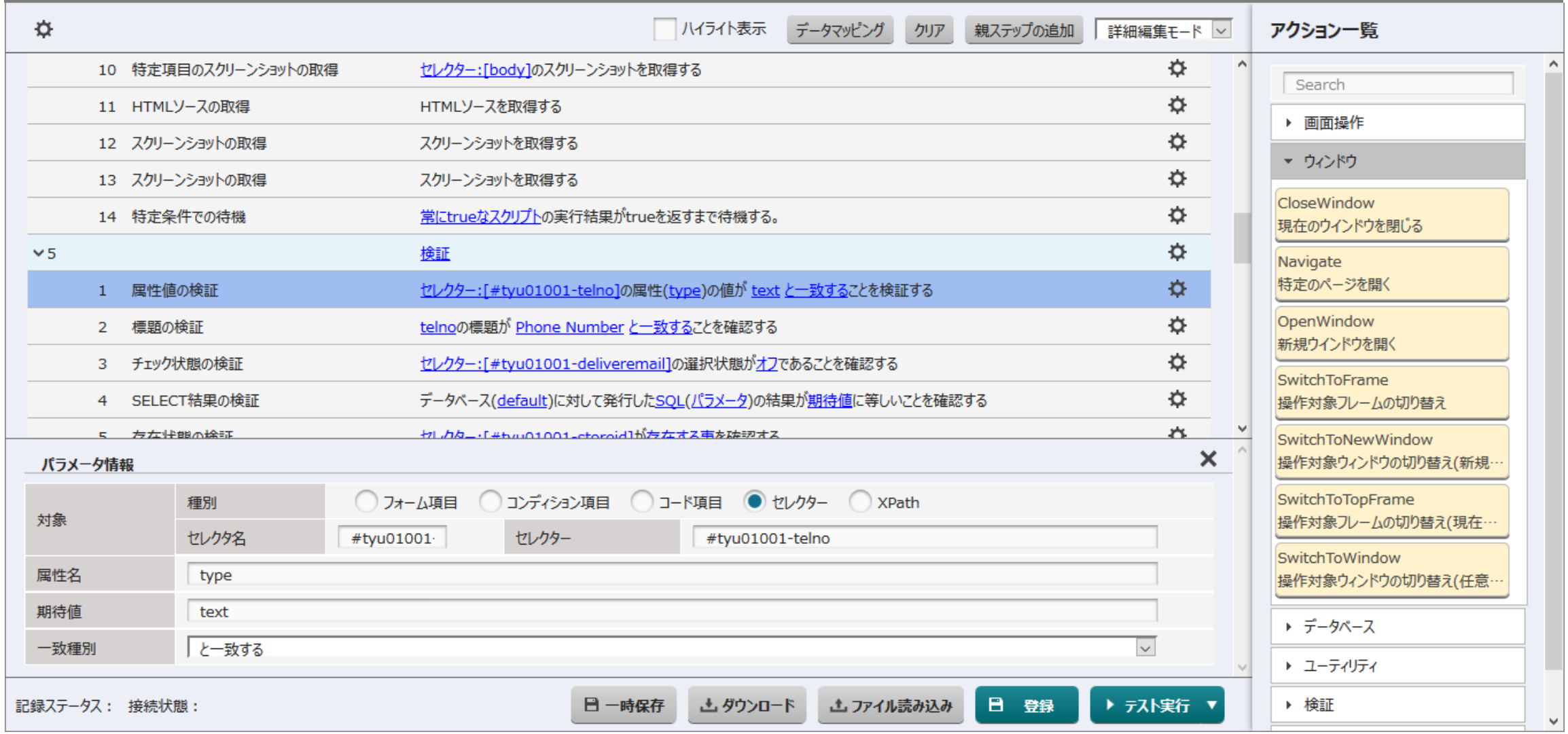
キーワード + パラメータ
による文章表現で可読性を確保

記録ステータス：停止中　接続状態：オフライン　一時保存　ダウンロード　ファイル読み込み　登録　テスト実行

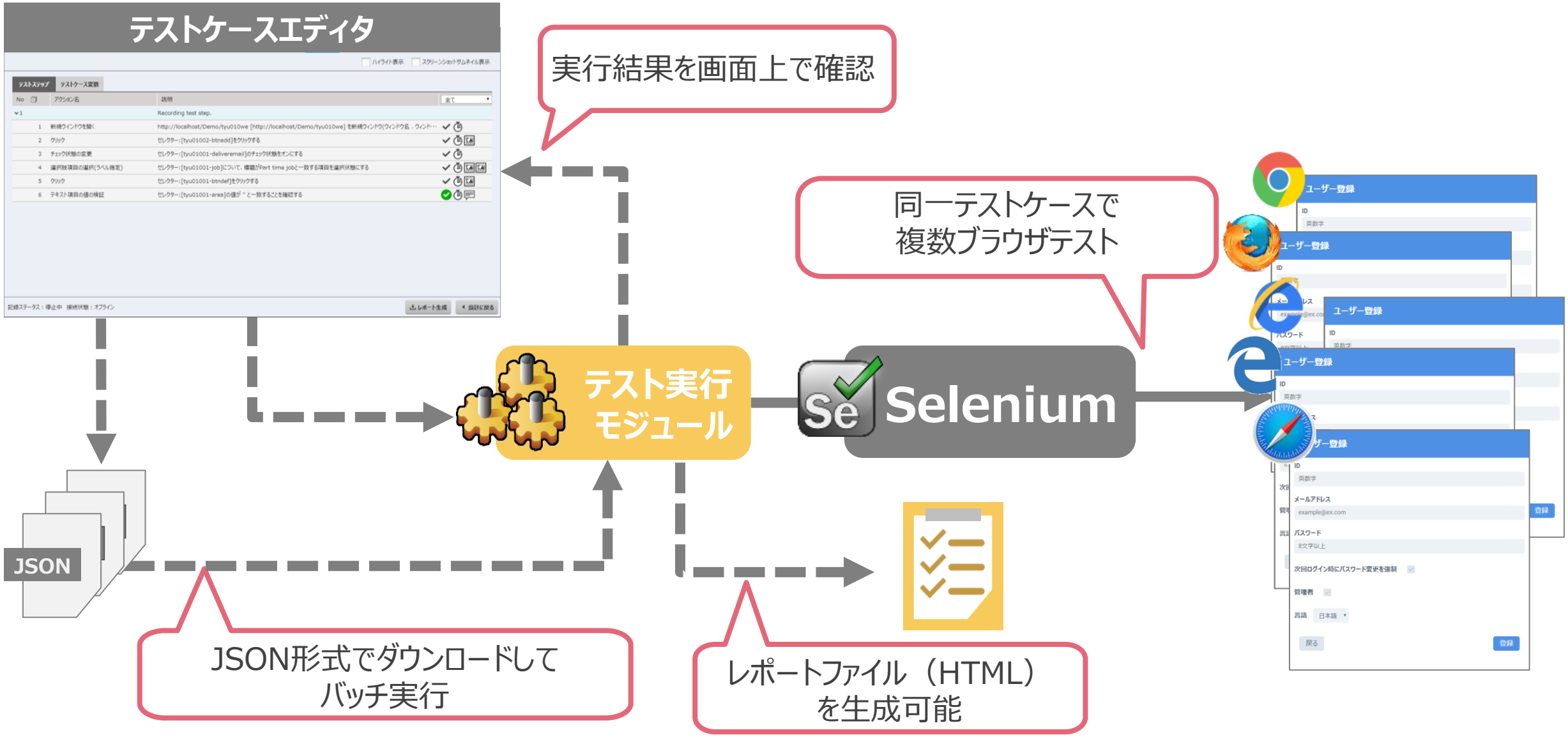
編集機能：「操作」の追加/変更/削除/並び替え/グループ化

FUJITSU

テストケースエディタ



テスト実行機能



結果評価・レポート

全体サマリー

No.	Test Case Id	Test Case Name	Start Date	End Date	Steps(Success/Total)	Operation
1	TYU010WE_001	顧客マスメン_ユーザ…	2016/10/21 18:23:43	2016/10/21 18:23:51	11/11	
2	TYU010WE_002	顧客マスメン_ユーザ…	2016/10/21 18:23:52	2016/10/21 18:23:59	8/9	

1ケース単位の結果

No	アクション名	説明	全て
1		画面を開く	
1	新規ウィンドウを開く	ユーザー登録 [http://localhost/Demo/tyu010we] を新規ウィンドウ(ウィンドウ名 、ウィンドウスタイル)で開く	
2	クリック	セレクター:[tyu01002-btnadd]をクリックする	
2		画面の値を入力する	
1	クリック	セレクター:[tyu01001-area]をクリックする	
2	テキスト項目の値の検証	セレクター:[tyu01001-area]の値が 東京都　大田区 と一致することを確認する	
3	クリック	セレクター:[tyu01001-email]をクリックする	
4	クリック	セレクター:[tyu01001-storecodget]をクリックする	
3		属性値の検証	
1	属性値の検証	セレクター:[tyu01001-email]の属性(id)の値が tyu01001-email と一致することを検証する	
2	属性値の検証	セレクター:[tyu01001-email]の属性(name)の値が tyu01001-email と一致することを検証する	
3	属性値の検証	セレクター:[tyu01001-email]の属性(class)の値が np-size-20 ifm-mode-enable np-require と一致することを検証する	
4	属性値の検証	セレクター:[tyu01001-email]の属性(placeholder)の値が example2@email.com と一致することを検証する	
5	属性値の検証	セレクター:[tyu01001-email]の属性(type)の値が text と一致することを検証する	
6	属性値の検証	セレクター:[tyu01001-email]の属性(value)の値が '' と一致することを検証する	

HTMLファイルなので
セットアップ不要

画面の値/エラーログ
/スクリーンショットなど
エビデンス自動採取

様々なテスト自動化を実現するための機能（一部抜粋）

FUJITSU

データマッピング機能

DBアクセス機能

スクリーンショット比較検証

[1] いろいろな値のテストを行う場合

例：「**ログインID**は**英数字以外**の場合にエラーになる事」をテストする

「ログインID」に**漢字**を入力するテスト	
1	「ユーザ登録」を開く
2	「ログインID」に「**富士通**」を入力する
3	「登録ボタン」をクリック
4	「メッセージ」が「**ログインIDに富士通は使えません**」が表示されていることを確認する

「ログインID」に**カタカナ**を入力するテスト	
1	「ユーザ登録」を開く
2	「ログインID」に「**アイウ**」を入力する
3	「登録ボタン」をクリック
4	「メッセージ」が「**ログインIDにアイウは使えません**」が表示されていることを確認する

[1] データマッピング機能

FUJITSU

テストケースエディタ

パラメータに
マッピング

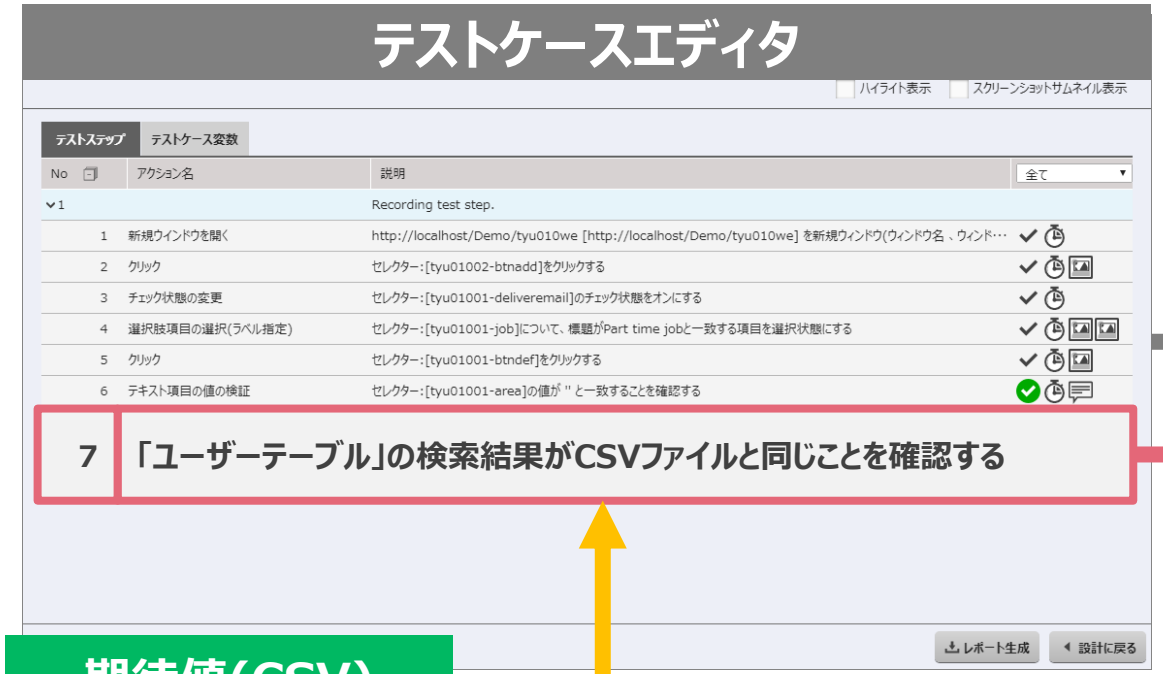
テストデータ（CSV）

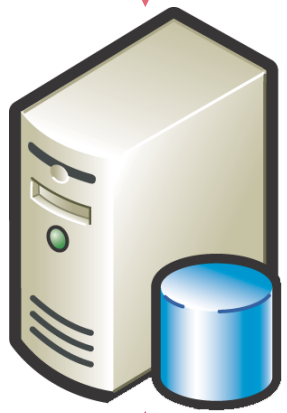
	A	B	C
1	パターン名	ログインID	メッセージ
2	漢字のテスト	富士通	ログインIDに富士通は使えません
3	ひらがなのテスト	あいう	ログインIDにあいうは使えません
4	カタカナのテスト	アイウ	ログインIDにアイウは使えません
5	半角カナのテスト	ｱｲｳ	ログインIDにｱｲｳは使えません
6	記号のテスト①	％％％	ログインIDに％％％は使えません
7	記号のテスト②	＃＃＃	ログインIDに＃＃＃は使えません
8	記号のテスト③	｜｜｜	ログインIDに｜｜｜は使えません
9	記号のテスト④	＆＆＆	ログインIDに＆＆＆は使えません
10			

ケース作成の手間半減

**複雑な組み合わせは
Excelで整理**

[2] DBアクセス機能





画面に見えないデータの確認

テスト前のデータ投入

[3] 画面デザインの変更をテストする場合

例：　画面デザインが「**変わってないか**」をテストする

[3] 画面デザインの変更をテストする場合

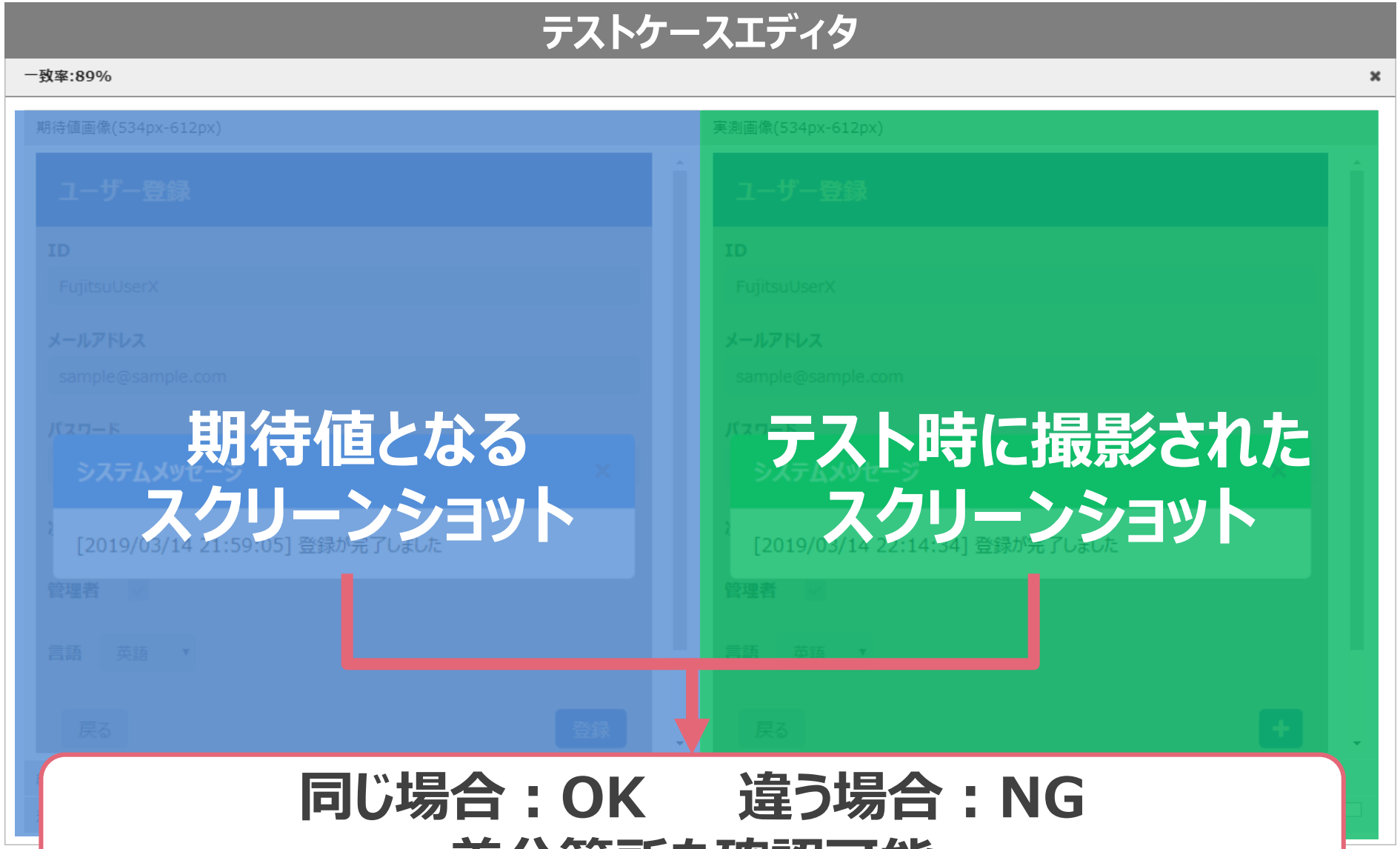
例：　画面デザインが「**変わってないか**」をテストする

修正前のアプリ

テスト対象アプリ

目視での確認は大変

[3] スクリーンショット比較検証：結果確認画面



[3] スクリーンショット比較検証：結果確認画面

[3] スクリーンショット比較検証：差分を黄色く強調表示

ヘッダーの色が違う
⇒強調表示

登録時間も本来は差分だが
対象外にも設定可能

デザインが違う
⇒強調表示

予期せぬ画面変更の検知

ブラウザバージョンアップの影響検知

【ご参考】 その他機能一覧

FUJITSU

豊富な「検証」機能

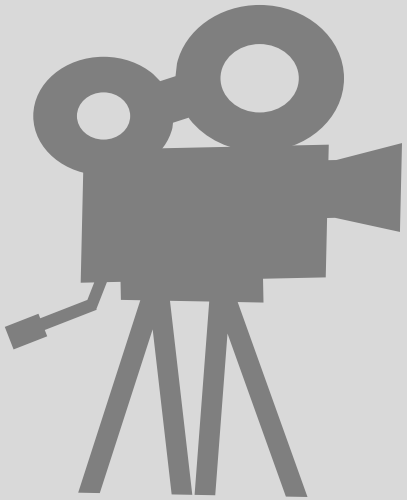
- テキスト項目の値
- チェック状態
- スタイル値
- 属性値
- フォーカス状態
- 表示状態
- 存在状態
- 選択肢項目（値）
- 選択肢項目（ラベル）
- 選択肢項目の選択要素（値）
- 選択肢項目の選択要素（ラベル）
- ウィンドウタイトル
- アラートダイアログメッセージ
- JavaScript実行結果
- シェルコマンドの実行結果
- DBに対するクエリの実行結果
- スクリーンショット

便利機能

- DB操作
  - CSVデータのINSERT
  - 汎用SQL実行
- HTMLソース取得
- スクリーンショット
  - 表示画面
  - 画面全体
  - 特定要素
- JavaScript実行
- シェルコマンド実行
- Sleep
- 変数の取り扱い

INTARFRM テストレコーディングツール

簡単に作れる

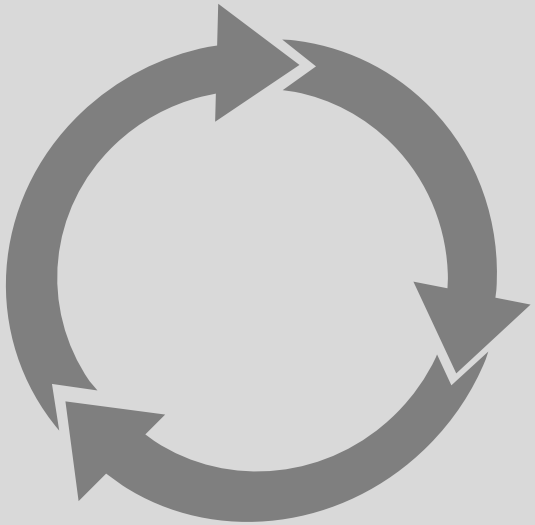


・操作の記録によるテスト設計
・GUI操作での多様なテスト設計
　・画面検証機能
　・DBテスト機能
　など

高いメンテナンス性

・操作の編集機能
・データ駆動テスト
・操作の共通化

長く使える

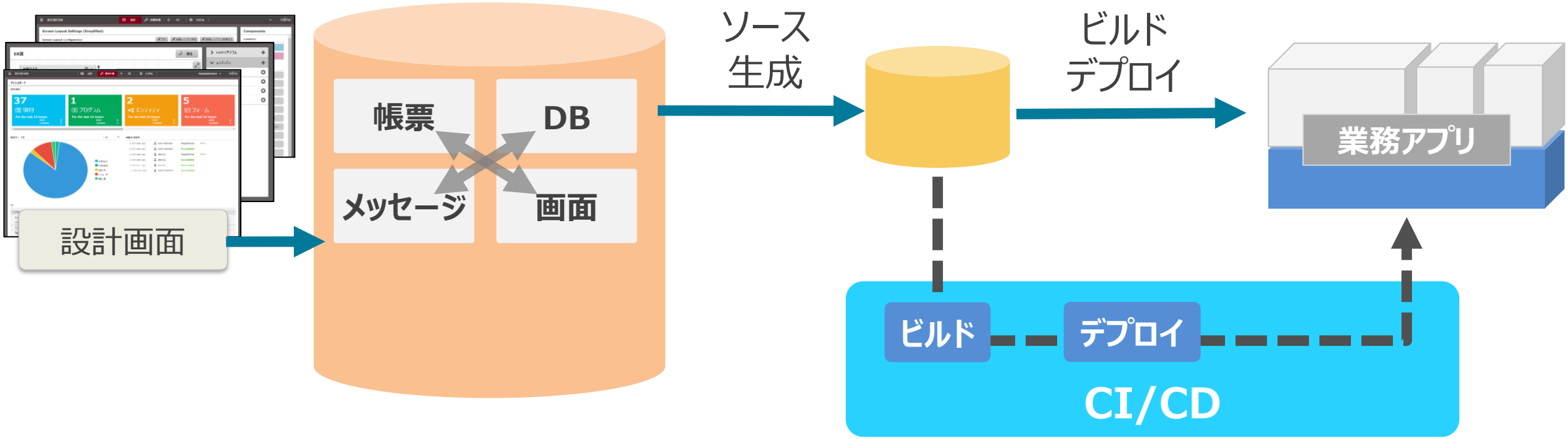


テスト設計 (JSON)
テストツール （Selenium)
の疎結合化

アプリケーション開発フレームワーク

INTARFRMの取り組み

INTARFRMを用いた設計・開発



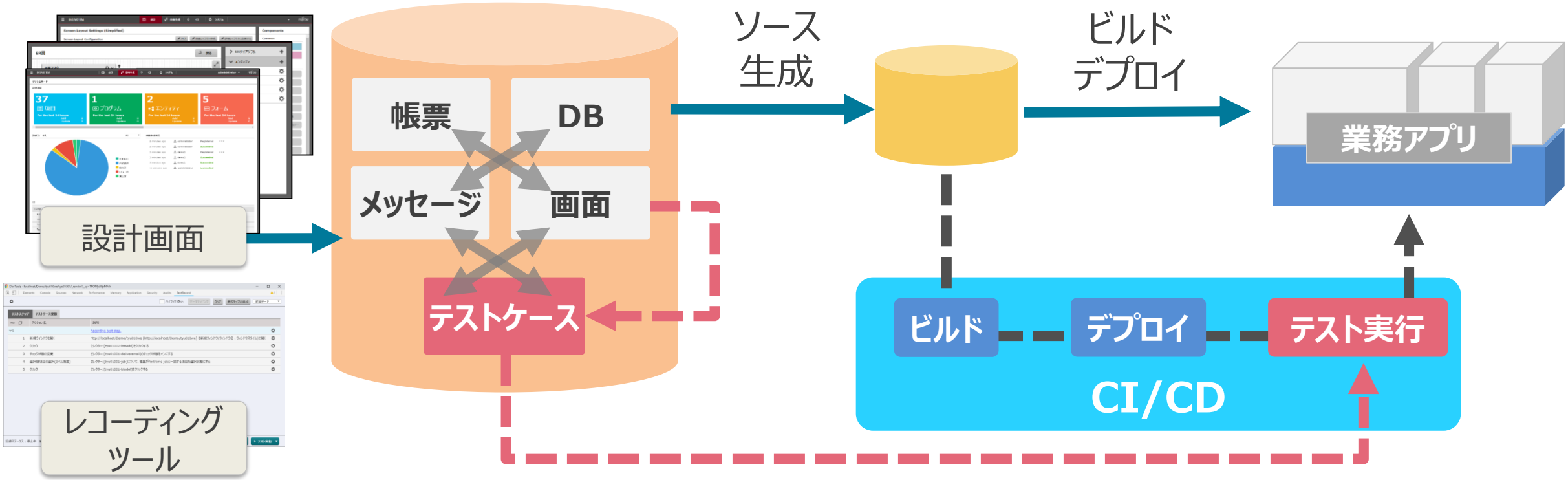
アプリケーション設計の集中管理

ソースコード自動生成

整合性チェック

CI/CD パイプライン実行

INTARFRMを用いた設計・開発・テスト



画面設計/テスト設計の集中管理

テストケース生成

テスト-画面設計 整合性チェック

CI/CD パイプライン実行

テスト自動化はどれくらい広まったか？

FUJITSU

弊社のテスト自動化状況 ・・・　　約10%

社内500人にアンケート

「自動化が進まない理由は何だと思いますか？」

テスト自動化はどれくらい広まったか？

FUJITSU

弊社のテスト自動化状況 ・・・　約10%

社内500人にアンケート

- 自動化のための時間とお金不足
- ツールや方法がわからない
- 自動化できる人材不足

テスト自動化はどれくらい広まったか？

FUJITSU

ツールを伸ばす

・対応範囲の拡大
・コスト削減の工夫
・AIの活用

プロセス標準を作成

・自動化作業のプロセス
・サンプル、ガイドライン

人材育成立上げ中

・教育、ワークショップ
・コミュニティ
・布教活動

自動化のための
時間とお金不足

ツールや方法が
わからない

自動化できる
人材不足

最後に

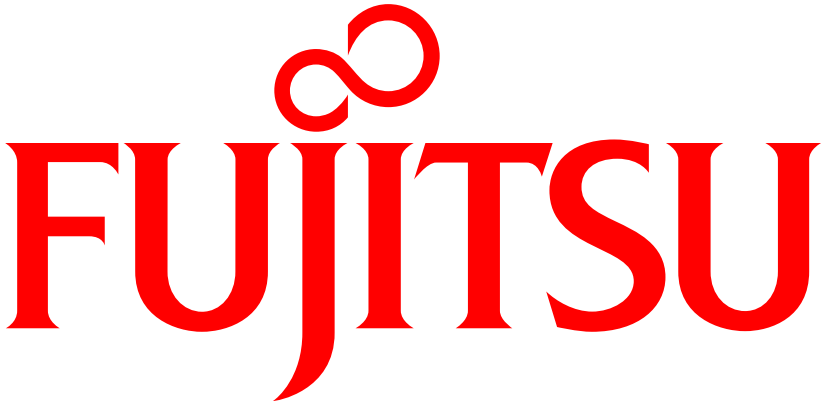
10年テスト自動化をやってきて

FUJITSU

他の人が理解できないテスト自動化は厳禁

今あるツールも使えなくなるかも

自動化したテストは
数年先にもメンテできますか？



shaping tomorrow with you