



Hewlett Packard
Enterprise

HPE Software

プロセス連携を踏まえた モバイルテスト自動化

日本ヒューレット・パッカード株式会社
ITマネジメントプリセールス本部 テクニカルコンサルティング部 小宮山 晃

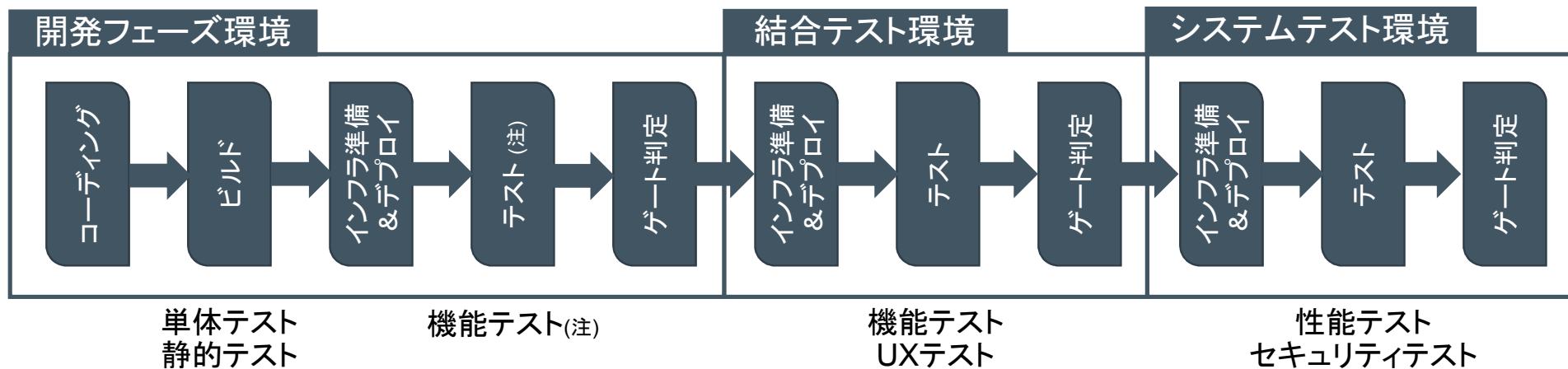
2016年2月25日 15:30 ~ 16:30

はじめに

- 本日の内容
 - プロセス連携の視点でモバイルテスト自動化について考慮するポイントを説明させていただき、どう工夫するかについてHPEソフトウェアでのソリューションをベースに説明します。
- 背景
 - テスト数の肥大化やさらなるコスト意識の強まりから、モバイルアプリケーションのテスト自動化についてここ数年お客様からの問い合わせが増えてきています。
 - モバイルアプリケーションのテスト自動化は、「どう自動化スクリプトを作成する事」も重要ではあるが、そこにフォーカスがされがちではないでしょうか？
 - 実際のテストフレームワークやDevOpsフレームワークにどう当てはめていくかも初期段階から考えておかないと部分最適で終わってしまう事になり、あてはめて連携させるための追加の工数やそのメンテナンス工数も増えてしまいます。

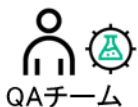
本日の資料で前提としているプロセスと主活動

テストに関わる「開発チーム」と「QA(テスト)チーム」の活動



開発チーム

- ・単体テスト実装(コンポーネント含む)
- ・UI機能テストの自動化実装(注)
- ・ゲート判定



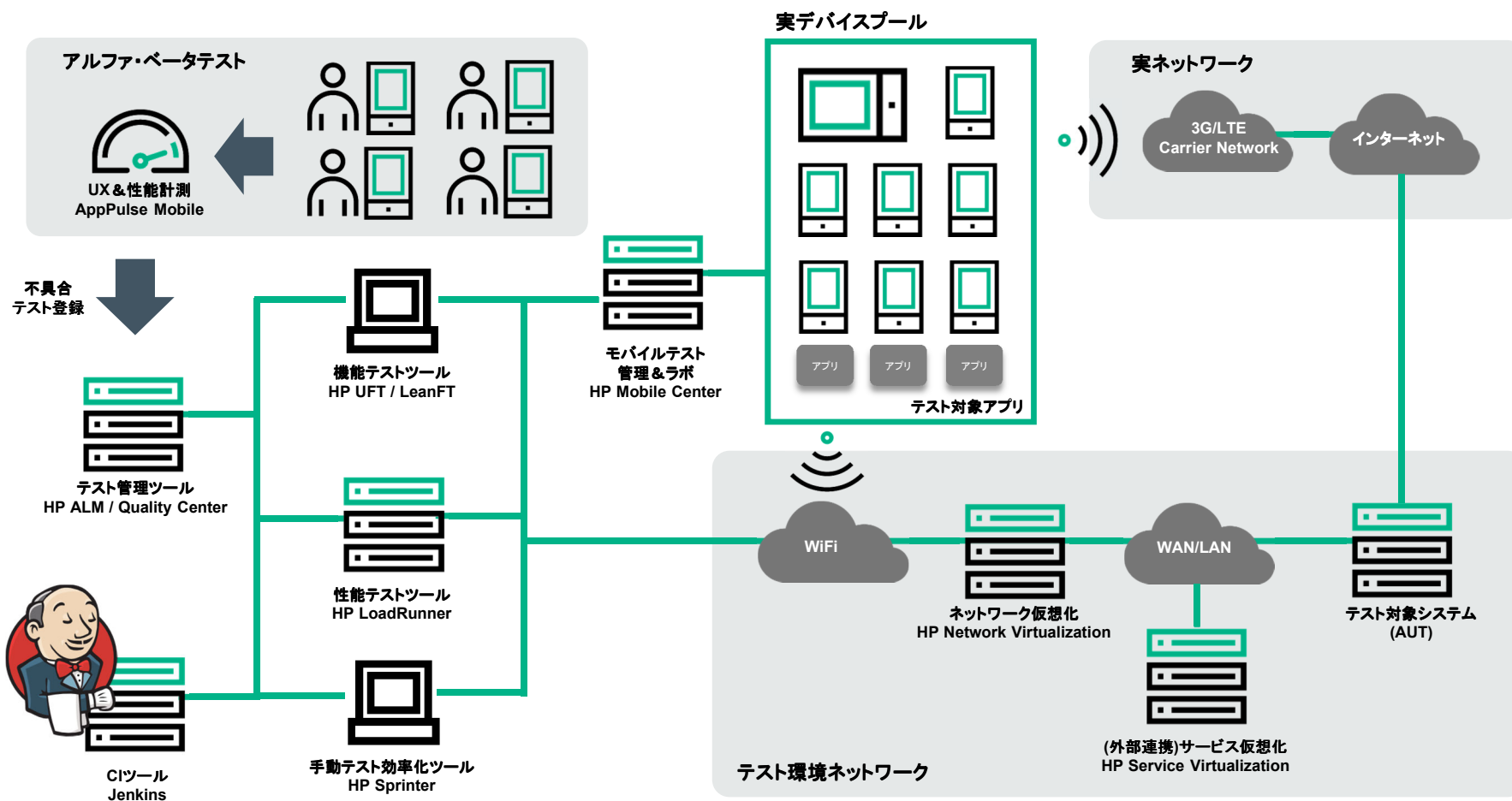
QAチーム

- ・テスト設計
- ・影響度分析>テストケース選定
- ・UI機能テストの自動化実装

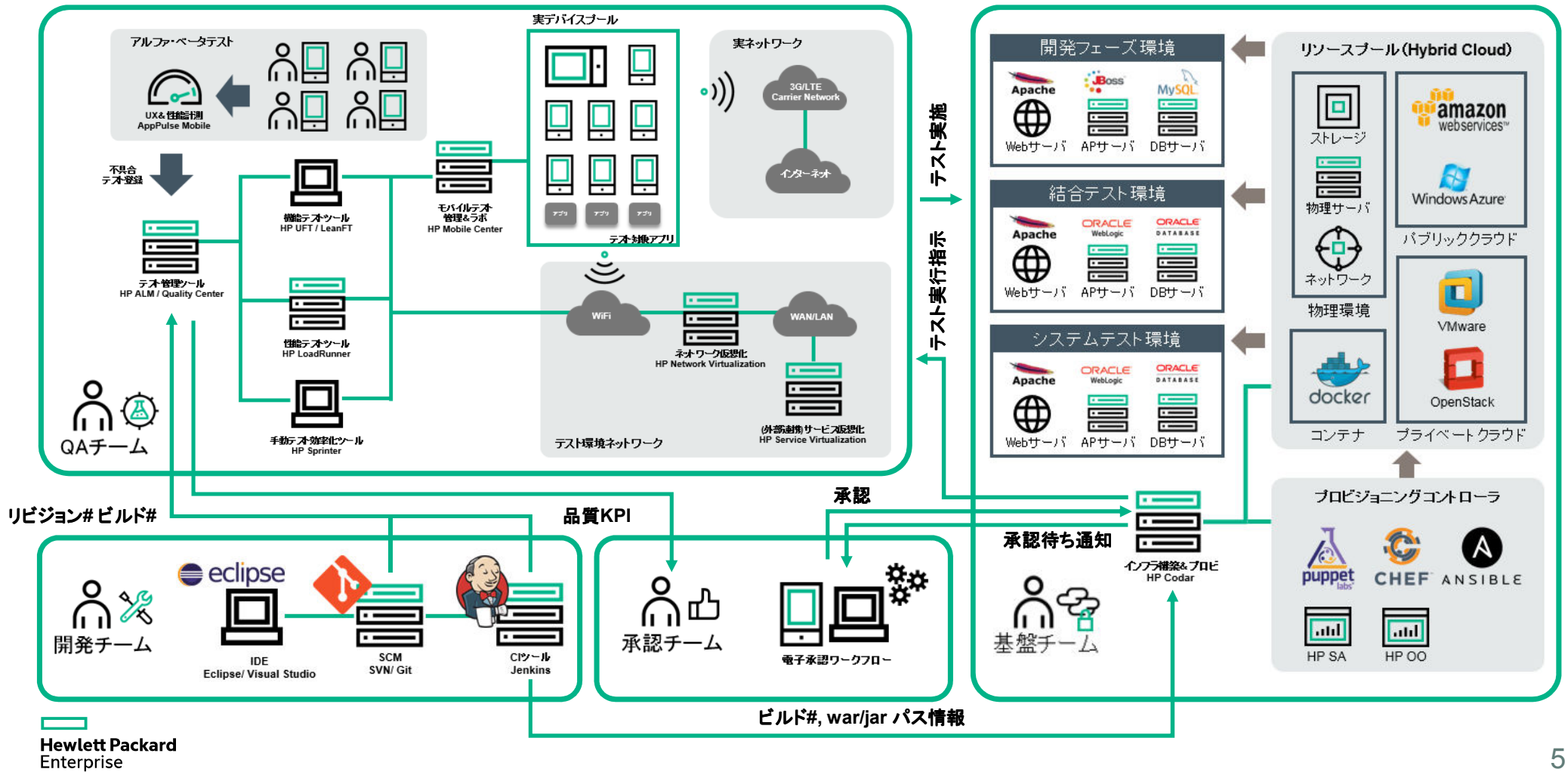
- ・手動テスト
- ・ゲート判定

- ・ゲート判定

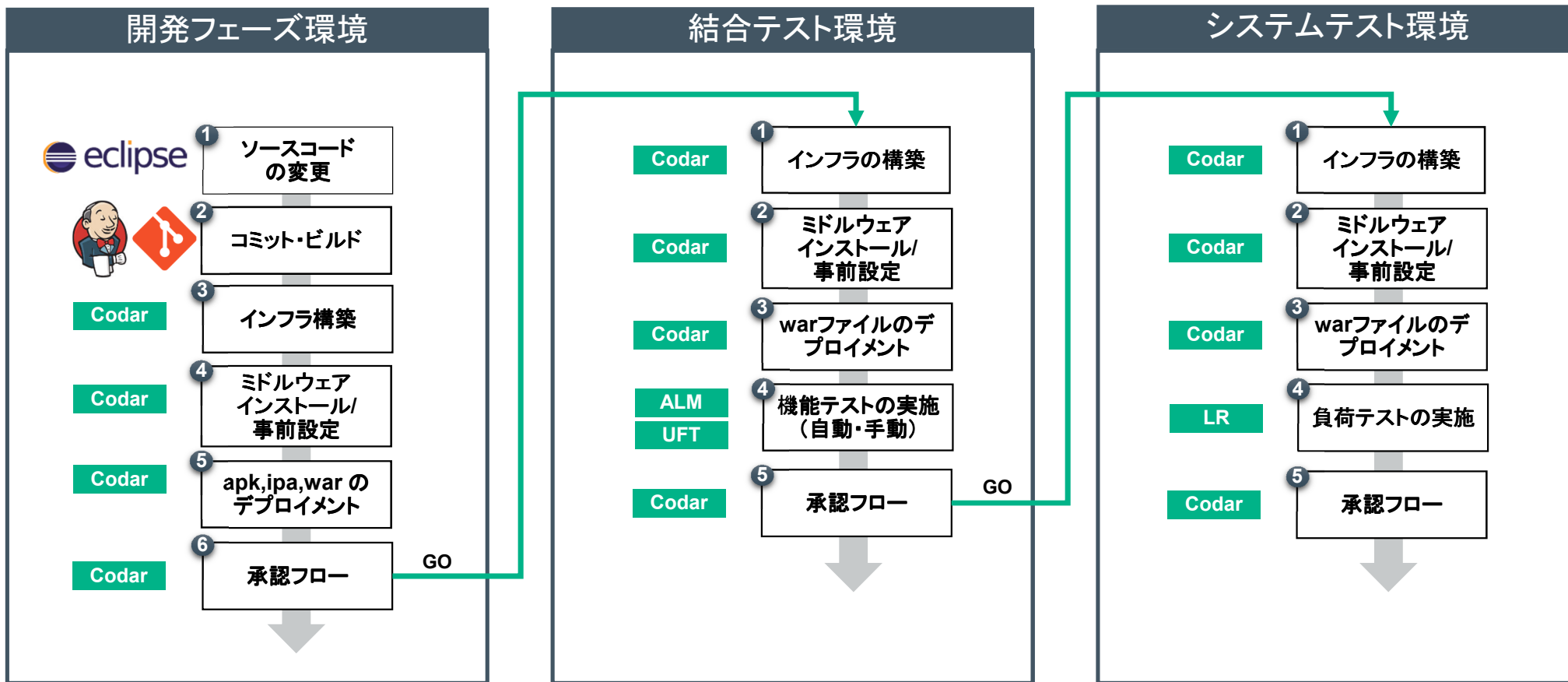
HPE ツールのモバイルアプリケーションテスト向け構成例



HPE モバイルDevOpsフレームワーク構成例(チーム単位)



前提プロセスを支援する HPE ツール 適用例



プロセス連携を踏まえた モバイルテスト自動化

1. 自動化取り組みの前に知っておく事と注意点
2. 自動化率について知るべきポイント
3. HPE UFT & HPE MC を使った自動化スクリプト作成手順
4. テスト用モバイルデバイス & アプリ管理(ラボ管理)
5. CI/CD自動化プロセスへの統合方法とTIPS
6. 複数デバイス & アプリケーションのコントロール方法とTIPS
7. インフラ構築 & デプロイ自動化ツールとテストツールの連携TIPS

自動化に取り込む前に

- 全てのテストケースが自動化できるわけではない
 - 自動化すべきテストケース、向かないテストケースを分別する

• 自動化に向いているもの

- 同一のシナリオに対して複数データでテストをする
- ビルド変更やパッチ適用の度にテストを繰り返す
- ビジネスに影響の大きい重要なアプリケーション
- 単純な操作
- 機能的に安定したアプリケーション

• 自動化を避けた方がよいもの

- 複雑な検証やシナリオから始める(処理毎に複数のテストに分けた方がよい)
- エラー処理や例外処理を1つのテストに含める(複雑な条件分岐が必要になる)
- 安定していない機能の自動テスト(仕様変更が頻発する箇所)
- すべてを自動化する
- 1回しか行わないテストの自動化

- 手動テストにすべきテスト活動に人的リソースを注力する



単機能確認を中心に、デバイスやOS、パッチなど組み合わせにより肥大化するテストケースに対して積極的に自動化適用率を上げ、デグレテストの工数削減を実現

自動化により効率化した人的リソースを、人によるスキルが求められるユーザビリティテスト、経験的・探索的テスト、音声・動画などのテストへ集中し品質向上に貢献

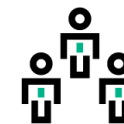


タイミングに依存する不具合(省電力モード切替瞬間での操作や複数機能の同時押下)など経験的テストは重要

モバイルアプリにおいて自動化が難しい操作（弊社経験上）

- 地図やチャットのように標示される内容が画像でしか判断できないもの（オブジェクトでのチェックができない、画像チェックも毎回異なる（地図座標やコンパスによるずれ））
- 通話機能操作（メッセージを話す、メッセージを聞き取る）
- カメラ（特定の画像、タイミングで撮影したものを確認する）
- ICカード機能（オサイフ機能など）
- センサー機能（振ってアプリが変わるとか、AR機能で標示されるとか、地図がコンパスで動くという操作）
- 複雑なジェスチャー操作（ジェスチャーを組み合わせたり軌跡が複雑な操作）

予めツールの不得意な操作を理解し、事前に該当テストケースについては手動テストで実施するようにリソースを調整する



手動

自動化範囲の向上にはプロセス改善も併せて実施

- 自動化範囲対象

- テストレベル **単体・コンポーネント、結合、システム、受入、運用**
- テストタイプ **機能、非機能(性能、セキュリティ、ユーザビリティ)**
- テストプロセス **計画、分析、設計、実装、実行、判定**

赤字部分が主にツールによる自動化が実践されている領域

取り組みのベストプラクティス

自動化

短期間で効果が
可視化しやすい

テストプロセス改善

効果が分かるまで
時間がかかりやすい

同時に取り組んで行く方がよい

テストプロセス改善の必要性はほとんどのチームで課題となっているが、時間がかかる事と可視化が難しいため投資がされにくい。自動化への取り組みと同時に進める事で社内の投資判断も得られやすい

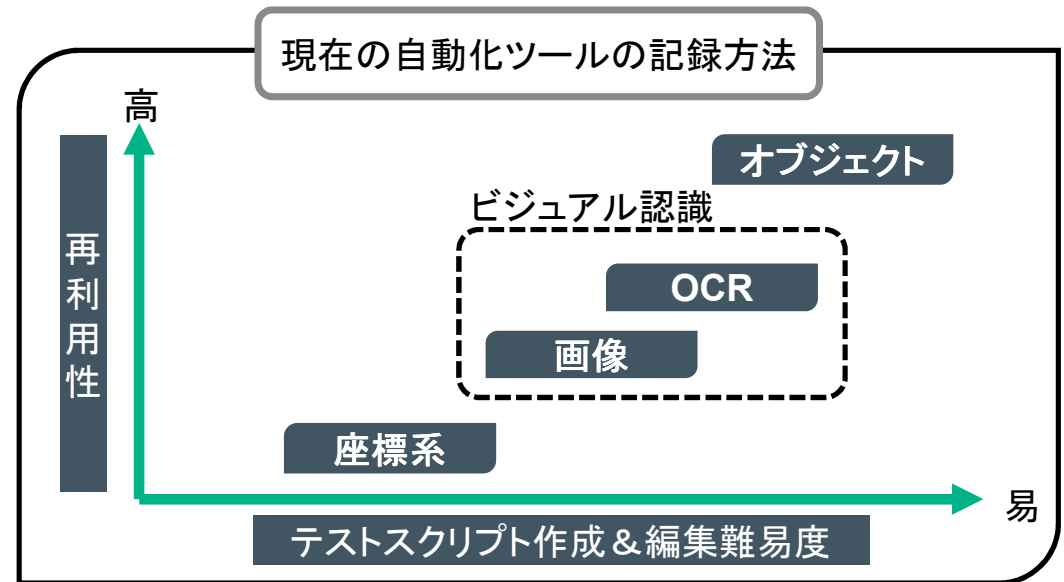
プロセス連携を踏まえた モバイルテスト自動化

1. 自動化取り組みの前に知っておく事と注意点
2. 自動化率について知るべきポイント
3. HPE UFT & HPE MC を使った自動化スクリプト作成手順
4. テスト用モバイルデバイス & アプリ管理(ラボ管理)
5. CI/CD自動化プロセスへの統合方法とTIPS
6. 複数デバイス & アプリケーションのコントロール方法
7. インフラ構築 & デプロイプロセスとテストツールの連携方法

モバイル機能テストの自動化率を支える自動化記録方法

- 自動化率①「自動化できたテストケース」/「自動化を予定していた全テストケース」を計測しておく
- 「自動化に向かないテストケース」を除いた自動化対象テストケースのうち、自動化できたテストケース率
- 自動化率が低い場合の要因として自動化ツールの技術的な制約が大きく、中でもモバイルアプリケーション記録方式(スクリプト化)は選定時の考慮ポイント

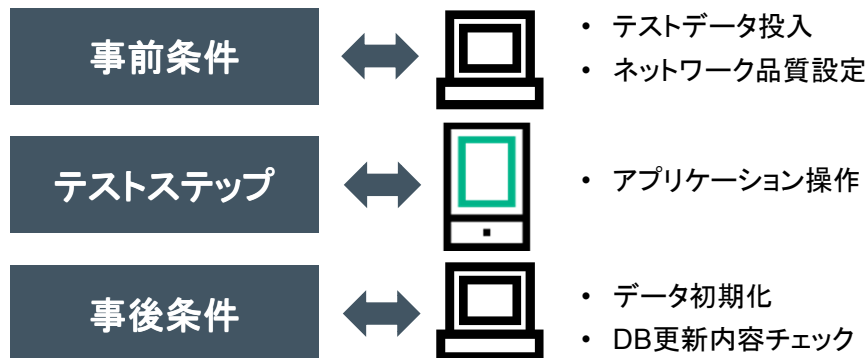
1. 座標系
座標軸(x, y)をベースに操作を記録します
2. ビジュアル
表示画面の情報からテキストをOCR、画像を対象オブジェクトとして操作を記録します
3. オブジェクト
ソースコードに専用ライブラリを利用しリコンパイル、または専用エージェント経由で対象アプリ操作する仕組みを利用し正確に記録します
ただしプリインストールアプリは対象外



モバイルアプリとはいえ、モバイルだけの自動化かどうか

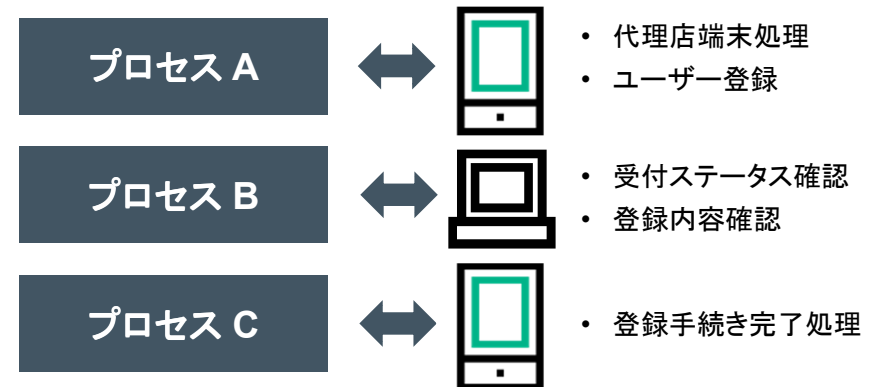
- 自動化率②「自動化できた手順」/「テストケース内で自動化を予定していた手順」を計測しておく
- 自動化率が低い場合の主な要因として自動化ツールの技術的な制約が考えられる

結合テストケース例



事前条件、事後条件で他システム操作が発生するテストケース

受入テストケース例



業務としてモバイル以外のシステム確認などが発生するテストケース

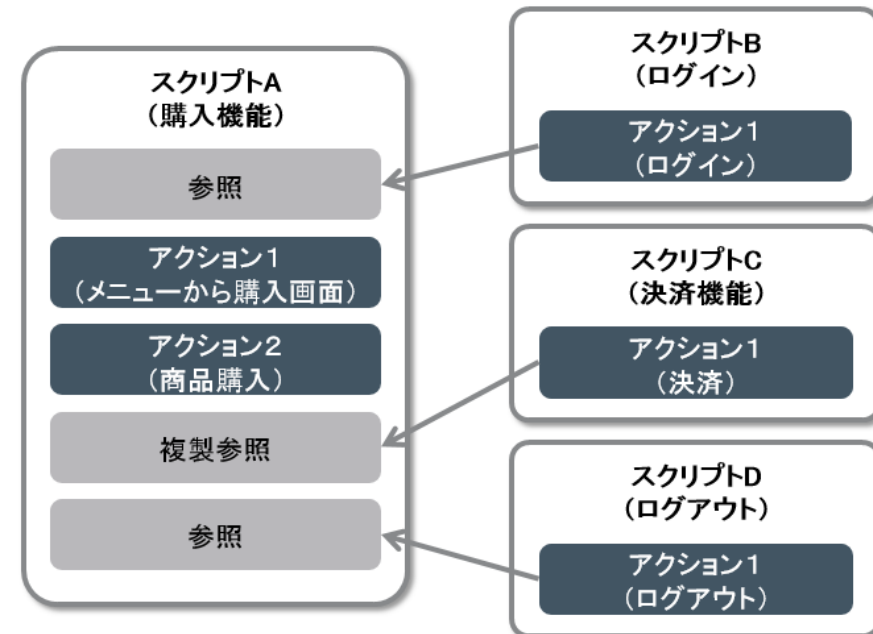
機能テストツールのコンポーネント活用

テストスクリプトの保守性を高め、モバイル以外の対応により自動化率を高率化

- 自動化に取り組んだものの保守性が低く回帰テスト工数が減らない経験はありませんか？

画面Aに修正がはいり、デグレ確認のため、回帰テストとして影響範囲の全スクリプトを実行したい。ここで、もしシナリオごとにスクリプトを作っていたら100シナリオを修正しなくてはならなくなってしまう？

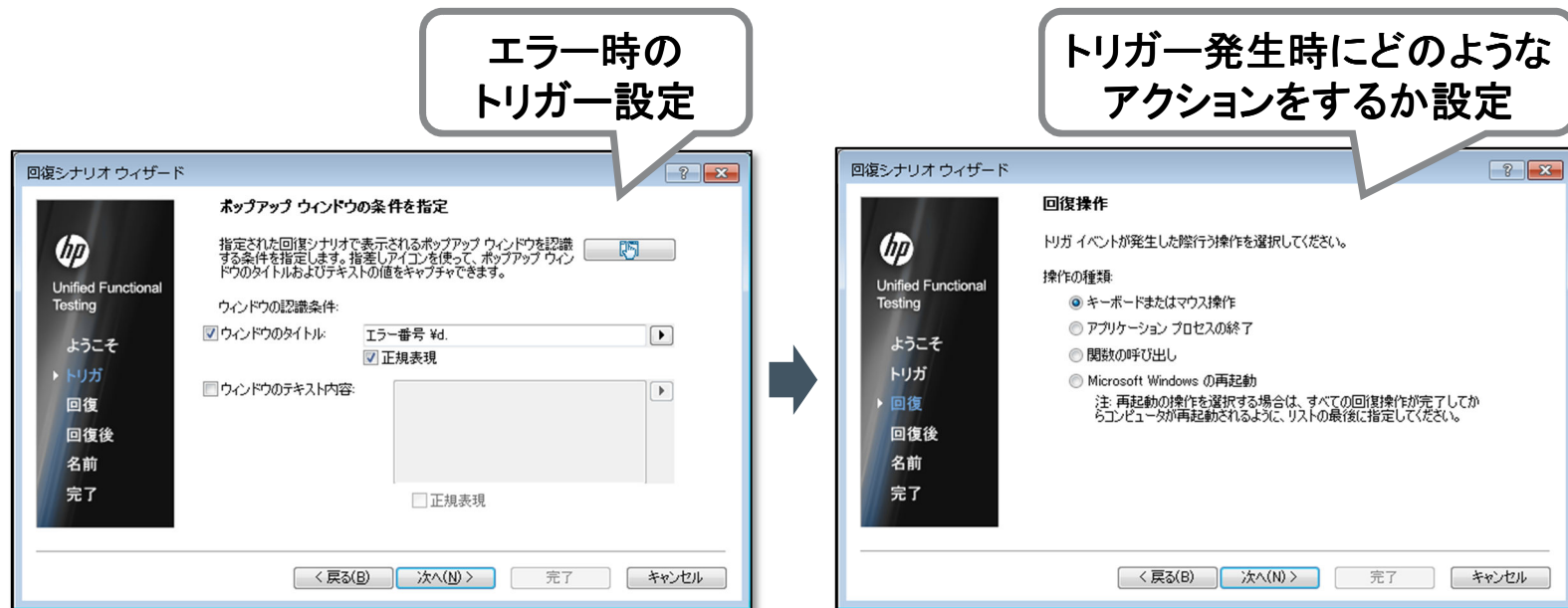
1. UFTでは、スクリプトを最小単位としてアクションとして管理、再利用する事が可能
2. 変更があった場合でもスクリプト修正に影響があるアクションのみに抑える事が可能
3. UFTではモバイルアプリだけではなく、Web、ERP (SAPやOracleEBS)、.NETアプリなども対応しておりテストの事前事後条件部分の処理や受入テスト時の複数端末を通したテストケースも可能



自動実行中に発生するエラー対処(夜間・週末実行時のエラーに対応)

発生するエラーから自動回復を支援

- 自動テストにおけるエラー対応処理は、テストスクリプト中にエラー処理をプログラミングしなければならないのが一般的な方法です
- UFTでは、「回復シナリオウィザード」に従ってエラーパターンを登録することで、プログラミングなしでエラー時の回復処理を追加できます

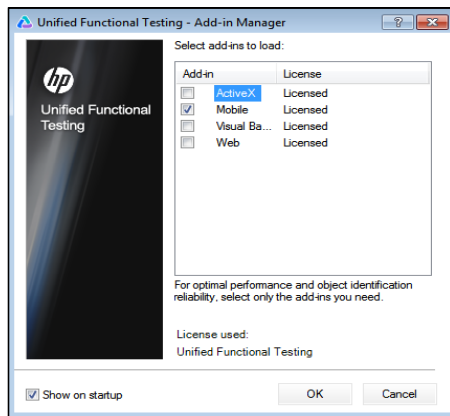


プロセス連携を踏まえた モバイルテスト自動化

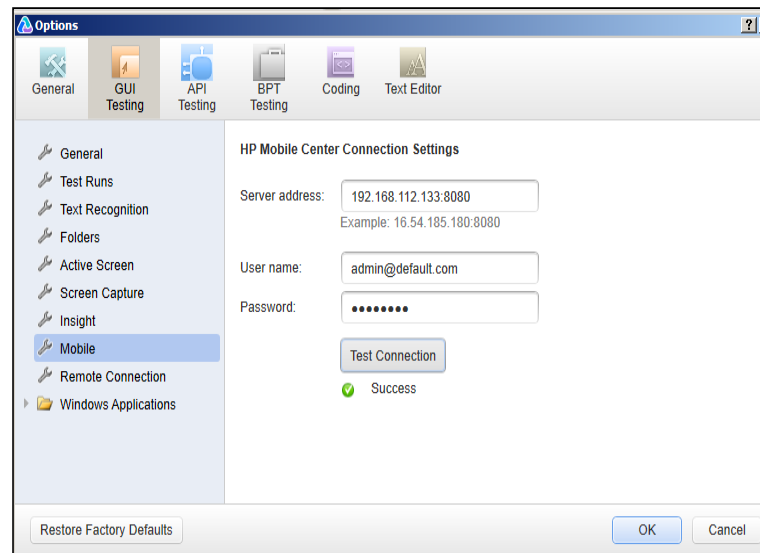
1. 自動化取り組みの前に知っておく事と注意点
2. 自動化率について知るべきポイント
3. HPE UFT & HPE MC を使った自動化スクリプト作成手順
4. テスト用モバイルデバイス & アプリ管理(ラボ管理)
5. CI/CD自動化プロセスへの統合方法とTIPS
6. 複数デバイス & アプリケーションのコントロール方法
7. インフラ構築 & デプロイプロセスとテストツールの連携方法

機能テスト操作 (1/3)

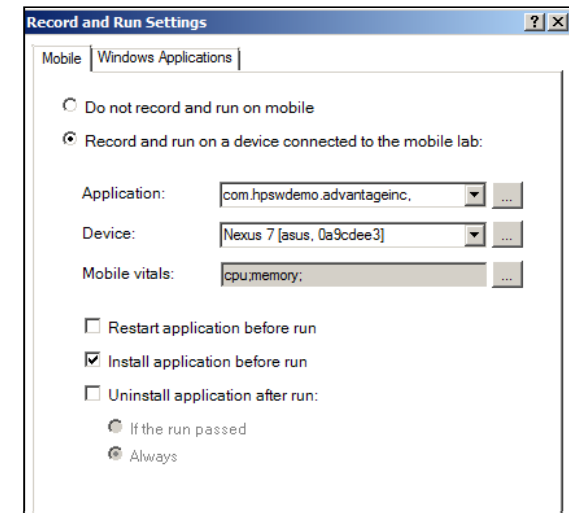
モバイルアプリケーション操作記録前準備



①UFTのアドインマネージャでMobileアドインを有効化



②オプションでMobile Center との接続情報やログインユーザ情報を入力



③実行環境設定でテスト対象アプリケーションや利用するデバイスを選択。CPUやMemory情報の取得設定やアプリケーションのデプロイ設定等も可能

機能テスト操作 (2/3)

モバイルアプリケーションの操作をRDPもしくは実デバイス操作でスクリプト記録

HP Unified Functional Testing - C:\Users\ADM\Documents\Unified Functional Testing\MobileCheck

ファイル(F) 編集(E) 表示(V) 検索(S) デザイン(D) 記録 実行 リソース ALM(A) ツール(T) ウィンドウ(W) ヘルプ(H)

ソリューション エクスプローラー

ソリューション 無題

MobileCheck

Action1

開始ページ

プロパティ

アクション設定

アクション名 Action1

場所 C:\Users\ADM\Documents\Unified Functional Testing\MobileCheck\Action1

場所 I:\Documents\Unified Functional Testing\MobileCheck\Action1

説明

再利用可能

```

1
2 *LOGIN Operation
3
4 Device("Device").App("Advantage_2").MobileEdit("MobileEdit").SetFocus
5 Device("Device").App("Advantage_2").MobileEdit("MobileEdit").Set "jojo"
6 Device("Device").App("Advantage_2").MobileEdit("MobileEdit_2").SetFocus
7 Device("Device").App("Advantage_2").MobileEdit("MobileEdit_2").Set "mercury"
8 Device("Device").App("Advantage_2").MobileButton("Login").Check CheckPoint("CheckPoint") 'Insert CheckPoint bas
9 Device("Device").App("Advantage_2").MobileButton("Login").Tap
10
11
12 *Select Money Transfer from Menu Pan
13
14 Device("Device").App("Advantage_2").MobileObject("Advantage.Open navigation drawer").Tap
15 Device("Device").App("Advantage_2").MobileLabel("Money transfer").Tap
16 Device("Device").App("Advantage_2").MobileEdit("MobileEdit").SetFocus
17 Device("Device").App("Advantage_2").MobileEdit("MobileEdit").Set "100"
18 Device("Device").App("Advantage_2").MobileButton("Transfer").Tap
19
20 Print 0 & ": " & Device("Device").App("Advantage_2").MobileView("class=Label", "index:=0").GetROProperty("text")
21
22 ActText = Device("Device").App("Advantage_2").MobileView("class=Label", "index:=0").GetROProperty("text")
23 'Read the expected value from the data table:
24 ExpText = DataTable.GlobalSheet.GetParameter("expText").Value
25
26 If ActText = ExpText Then
27 Reporter.ReportEvent micPass, "Verify Text", "Text is as expected: " & ExpText
28 Else
29 Reporter.ReportEvent micFail, "Verify Text", "Text is not as expected: " & ExpText
30 End If
31
32 Device("Device").App("Advantage_2").MobileButton("OK").Tap
33
34 Device("Device").App("Advantage_2").MobileObject("Advantage.Open navigation drawer").Tap

```



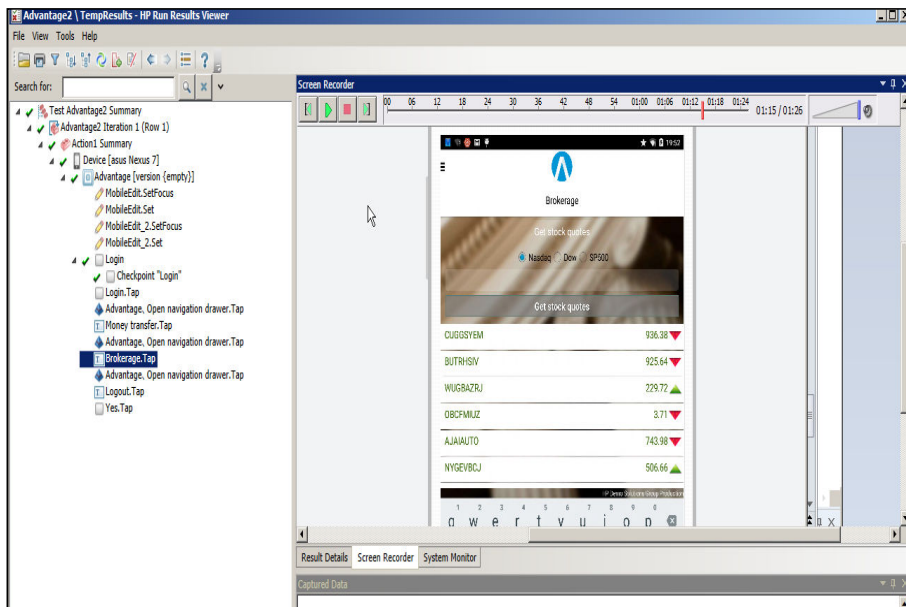
④アプリケーションのログイン操作をオブジェクトとして記録
どのボタンをタップしたか
フィールドに何を入力したか
等を個別オブジェクトとして記録

⑤UFTのスクリプトとして記録実行が可能

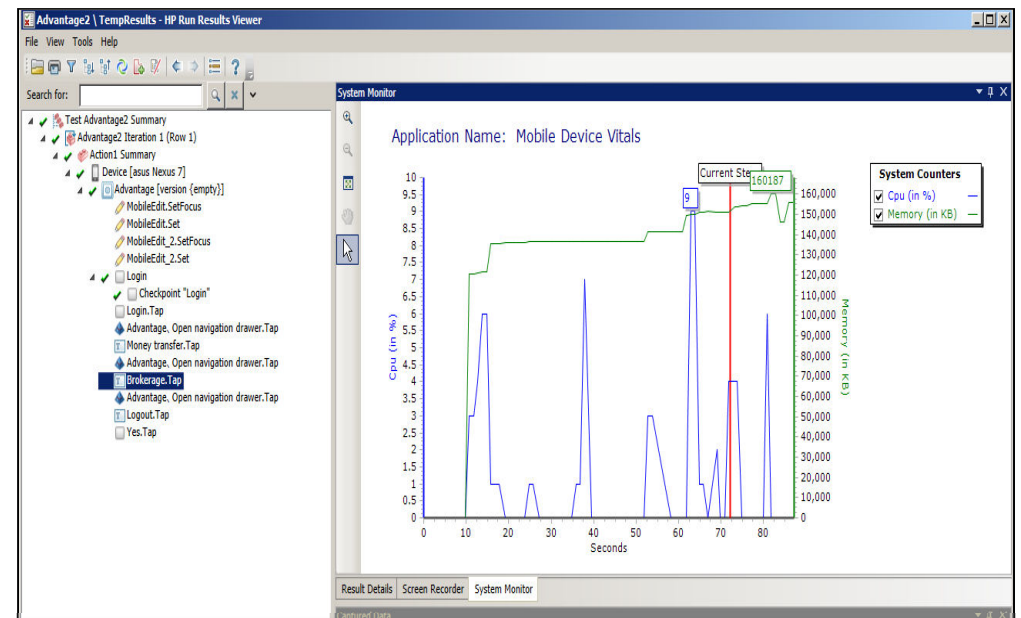
オブジェクトとして認識され、チェックポイントやデータ駆動型のテストも実施可能

機能テスト操作 (3/3)

テスト結果レポート(チェックポイント結果、エビデンス画面 & 動画、デバイスバイタル & ログ)



⑥各操作ステップの画像(エビデンス)や実行動画を取得しテスト実行時のデバイス動作を確認することも可能



⑦デバイス側のCPUやメモリ状況などを実行ステップと関連しながら確認することが可能(注)

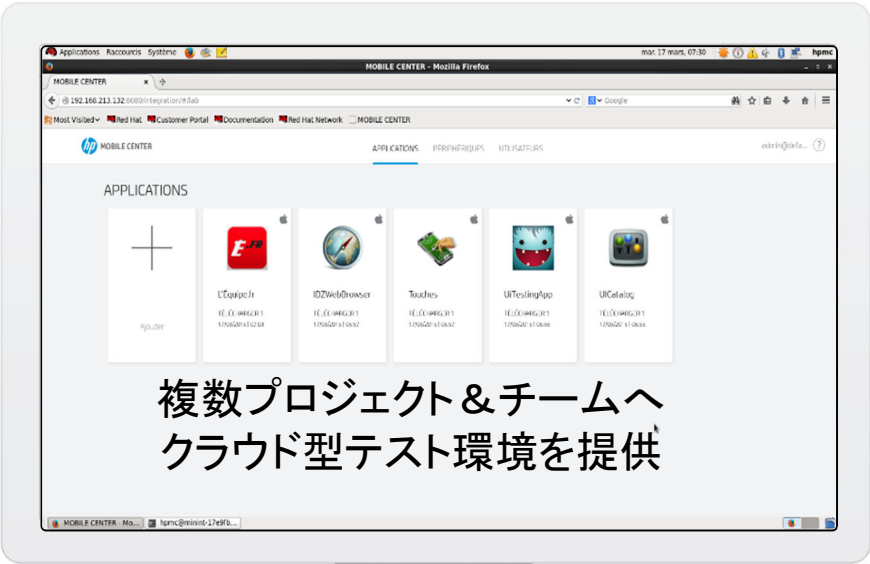
どのステップでCPU消費量が高くなるのか等を分析することが可能

プロセス連携を踏まえた モバイルテスト自動化

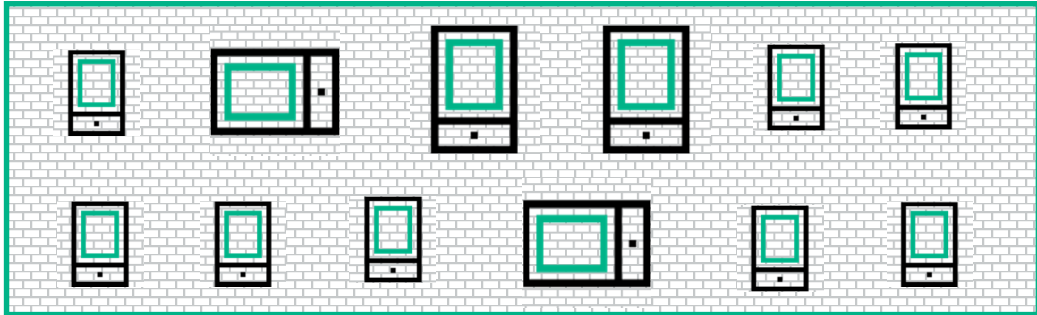
1. 自動化取り組みの前に知っておく事と注意点
2. 自動化率について知るべきポイント
3. HPE UFT & HPE MC を使った自動化スクリプト作成手順
4. テスト用モバイルデバイス & アプリ管理(ラボ管理)
5. CI/CD自動化プロセスへの統合方法とTIPS
6. 複数デバイス & アプリケーションのコントロール方法
7. インフラ構築 & デプロイプロセスとテストツールの連携方法

「管理 & ラボ機能」により大規模で煩雑になりがちな管理に対応 テスターはUFTからデバイスとアプリをクラウドとして利用する事で管理から開放されます

ラボ管理機能(HP Mobile Center)



デバイスとOSの管理



オンプレミス, iOS, Android, プラグ & プレイ

アプリの管理



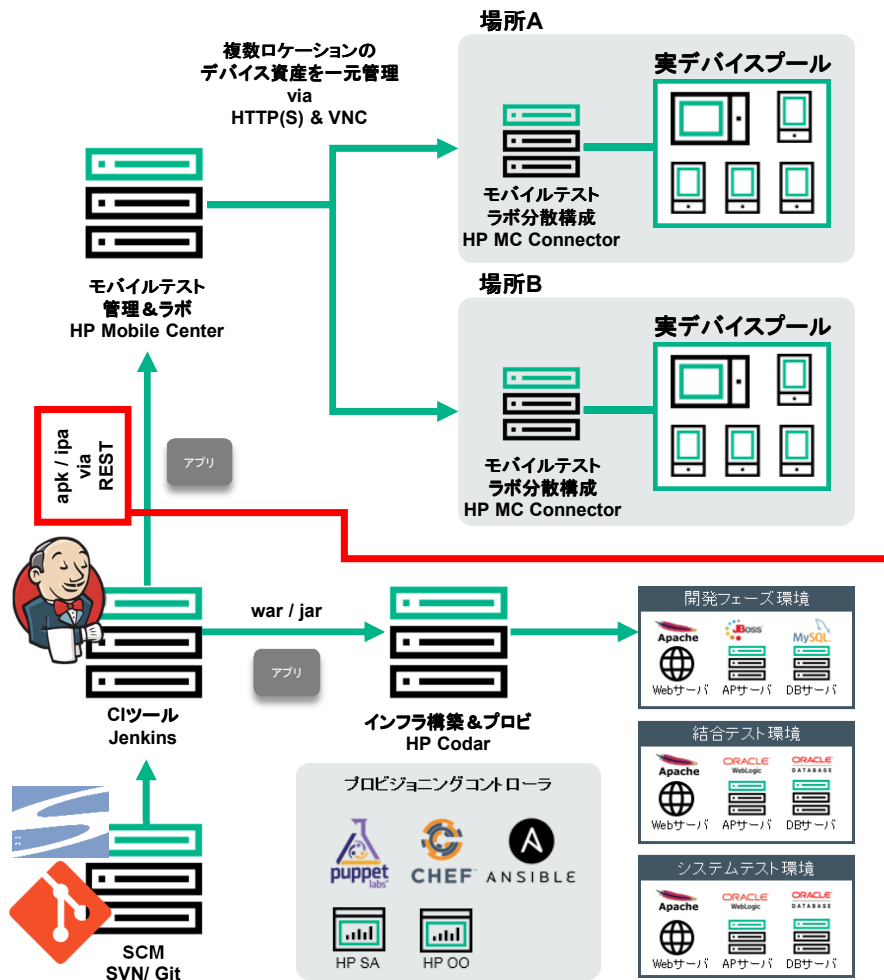
配布, インストール, アンインストール

CIツール連携し apk/ipa ファイルをデプロイ、インストール可能

プロセス連携を踏まえた モバイルテスト自動化

1. 自動化取り組みの前に知っておく事と注意点
2. 自動化率について知るべきポイント
3. HPE UFT & HPE MC を使った自動化スクリプト作成手順
4. テスト用モバイルデバイス & アプリ管理(ラボ管理)
5. CI/CD自動化プロセスへの統合方法とTIPS
6. 複数デバイス & アプリケーションのコントロール方法
7. インフラ構築 & デプロイプロセスとテストツールの連携方法

テスト時のアプリデプロイ自動化 (1/2)



REST サンプル (Ruby ケース) : このスクリプトを CI ツールから実行

```

import sys, json, requests
PROTOCOL = 'http'
PORT = '8080'

SERVER = sys.argv[1]
USER = sys.argv[2]
PASS = sys.argv[3]
FILE = sys.argv[4]

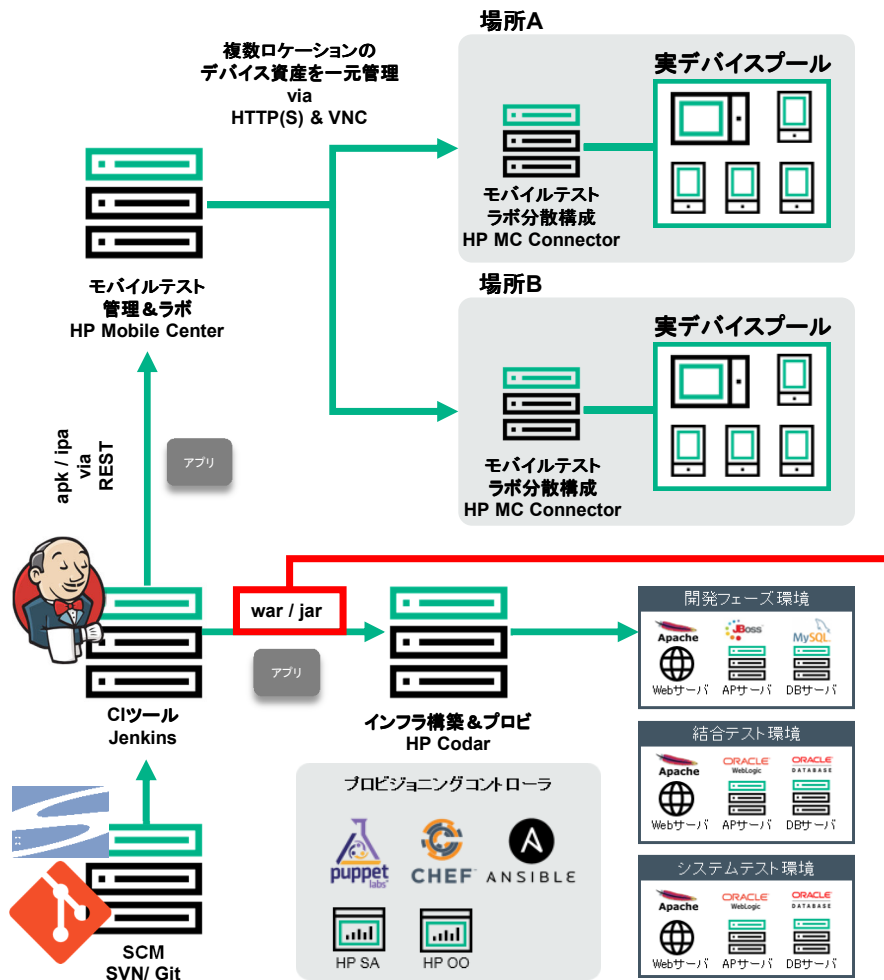
BASE_URL = PROTOCOL + '://' + SERVER + ':' + PORT + '/rest'
s = requests.Session()
# LOGIN
url = BASE_URL + '/client/login'
headers = {'Accept': 'application/json', 'Content-Type': 'application/json; charset=UTF-8'}
payload = {'name': USER, 'password': PASS, 'accountName': 'default'}
response = s.post(url, headers=headers, data=json.dumps(payload))
print ("Connect: ", response.status_code)

hp4msecret = response.headers['x-hp4msecret']
jsession = s.cookies['JSESSIONID']

if response.status_code == requests.codes.ok:
# UPLOAD APP
url = BASE_URL + '/apps'
file = {'fileUpload': open(FILE, 'rb')}
headers = {'x-hp4msecret': hp4msecret, 'JSESSIONID': jsession}
response = s.post(url, headers=headers, files=file)
print ("Upload: ", response.status_code)

if response.status_code == requests.codes.ok:
print ("App Name:", response.json()["name"])
print ("App Id:", response.json()["id"])
print ("App version:", response.json()["version"])
print ("App Count:", response.json()["count"])
    
```

テスト時のアプリデプロイ自動化 (2/2)



具体的な Codar 情報設定例

HP Codar Plugin
CodarUri*

Username*

Password*

SSLCertificatePath*

CertificatePassword*

Validate REST API Access

Continuous Promote

Enable Http Authentication

HttpUsername

HttpPassword

Application Design Location*

Environment*

Package Description

Package Properties*

Component Id/Name	Property Name	Property Value	
PetClinic DB C	configurationurl	mysqlqdb_conf.s	削除
PetClinic Applix	artifacturl	petclinic.war	削除
PetClinic Applix	configurationurl	spring_petclinic	削除

Jenkins から Codar へ連携するためのアクセス先、ID/PW 情報

概念モデル(注)の設定ファイルの存在するロケーションを指定

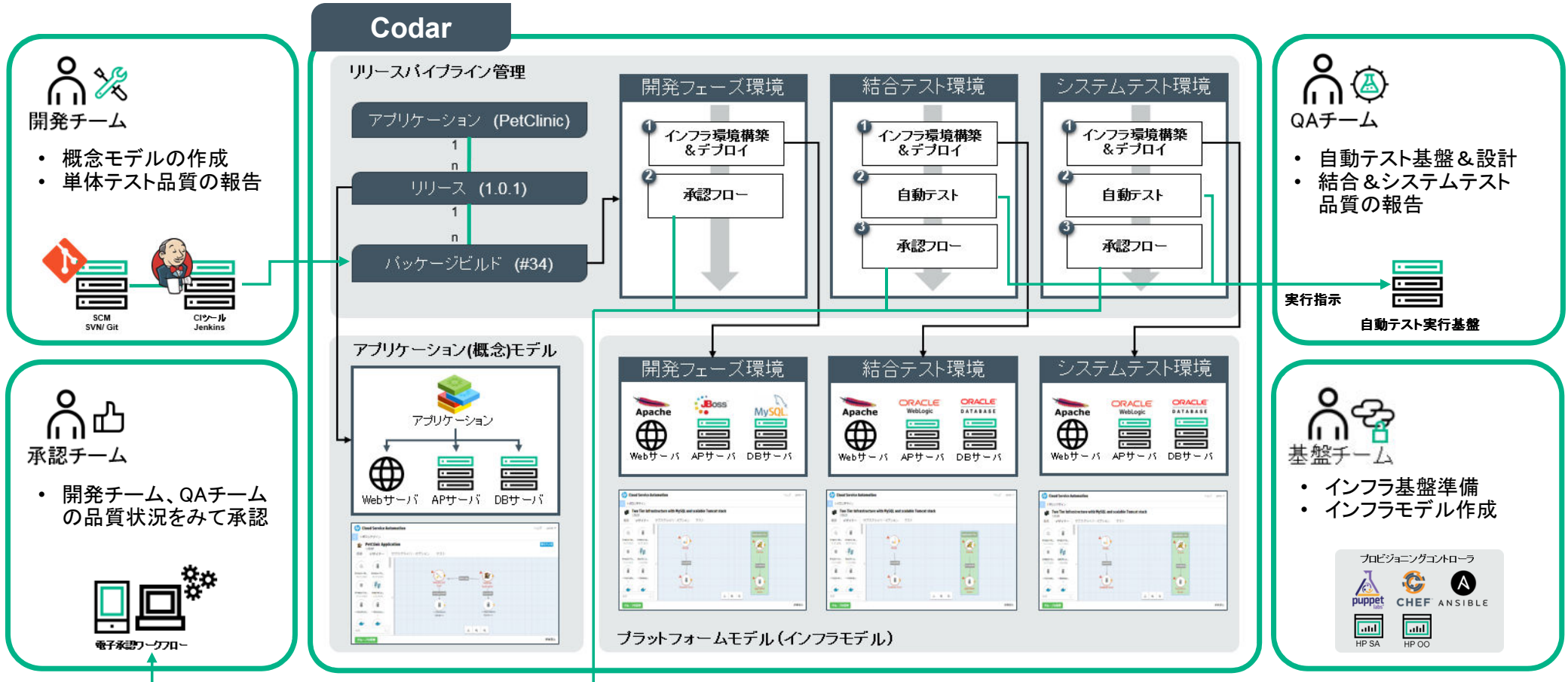
Codar からどの環境でデプロイするかをグループを設定

ビルド毎に変わる変数をCodarに渡す場合に利用

(注) 概念モデルについては次スライドで説明

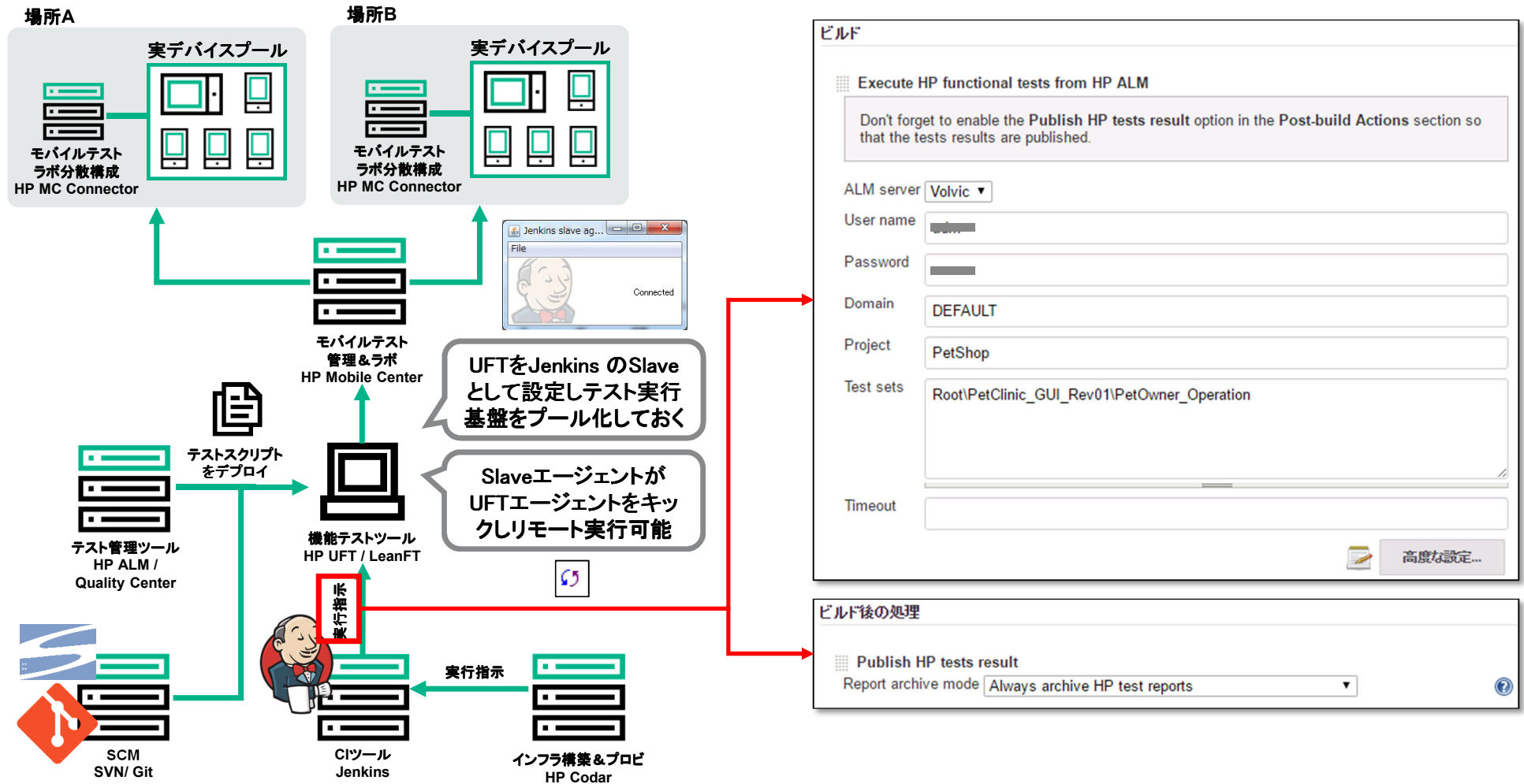
Codar をチーム全体で共有しプロセス標準化を実現

過去にデプロイした環境の開発からテストまで含めて保存し、短時間での環境再現が可能



CI/CDプロセスにおけるUFT実行 (1/3)

Jenkins専用アドインを利用して設定可能



CI/CDプロセスにおけるUFT実行 (2/3)

テスト結果はUFTやALMクライアントを利用せずにJenkinsでも確認することが可能

プロジェクト UFT_ALM_Script_2

説明を記入

プロジェクトの無効化

ワークスペース

変更履歴

最新のテスト結果 (1個の失敗 / +1)

テスト結果の推移

永続リンク

最新のビルド (#20) 16 秒 前

テスト結果

0個の失敗 (-1)

1個のテスト (±0)
所要時間 27 秒

説明を記入

すべてのテスト

パッケージ	テスト所要時間	失敗	(差分) スキップ	(差分) パス	(差分) 合計	(差分)
All-Tests	27 秒	0	-1	0	1	+1

コンソール出力

```

Started by user anonymous
Building remotely on UFT_Slave (UFT_Slave_1) in workspace
C:\Users\adm\Desktop\workspace\UFT_ALM_Script_2
[UFT_ALM_Script_2] $ C:\Users\adm\Desktop\workspace\UFT_ALM_Script_2\HpToolsLauncher.exe -paramfile
props1502201616165848058.txt
"Started..."
Timeout is set to: -1
Run mode is set to: RUN_REMOTE
---
15/02/2016 16:58:53 Running: [1]PetClinic_Demo_01
15/02/2016 16:58:53 Running test: [1]PetClinic_Demo_01, Test id: 2, Test instance id: 2
Test: [1]PetClinic_Demo_01, Id: 2, Execution status: Running
Test: [1]PetClinic_Demo_01, Id: 2, Execution status: Passed, Message: Test run passed.
Step: Start Test
      Start Test PetClinic_Demo_01
Step: Start Global Iteration
      Start Global Iteration 1
Step: Start Action
      Start Action Action1
Step: Verification, Status: Passed
      標準チェックポイント "Welcome!":
Step: Verification, Status: Passed
      ビットマップ チェックポイント "pets"
Step: Verification, Status: Passed
      標準チェックポイント "Find Owner":
Step: End Action
      End Action Action1
Step: End Global Iteration
      End Global Iteration 1
Step: End Test
      End Test PetClinic_Demo_01
Link: td://PetShop.DEFAULT.192.168.3.12:8080/qcbin/TestLabModule-000000003649890581?
EntityType=IRun&EntityID=13
15/02/2016 16:59:43 Test complete: [1]PetClinic_Demo_01, Run ID: 13
-----
Test: [1]PetClinic_Demo_01, Status: Passed, Message: Test run passed.
Link: td://PetShop.DEFAULT.192.168.3.12:8080/qcbin/TestLabModule-000000003649890581?
EntityType=IRun&EntityID=13
-----
Test set: PetOwner_Operation, finished at 15/02/2016 16:59:44
-----
Run status: Job succeeded, total tests: 1, succeeded: 1, failures: 0, errors: 0
Passed : Root\PetClinic_GUI_Rev01\PetOwner_Operation#[1]PetClinic_Demo_01
-----
Finished: SUCCESS
    
```

CI/CDプロセスにおけるUFT実行 (3/3)

ALM上からテスト結果と不具合のKPIやレポートをチェック可能 (QA担当者や承認者が利用)

The screenshot shows the HP Application Lifecycle Management interface. The top navigation bar includes 'テスト実行' (Test Execution), 'テストセット実行' (Test Set Execution), and 'ビルド検証スイートの実行' (Build Verification Suite Execution). The main area displays a table of test runs with columns for '実行ID' (Run ID), '実行名' (Run Name), 'テスト: テスト名' (Test: Test Name), '設定: 名前' (Configuration: Name), 'ステータス' (Status), '状態' (State), '経過時間' (Elapsed Time), and '実行日' (Run Date).

実行ID	実行名	テスト: テスト名	設定: 名前	ステータス	状態	経過時間	実行日
12	Run_2-9_10...	PetClinic_Demo...	PetClinic_Demo...	Failed		13	2016/02/09 19:39
11	Run_2-9_7.4...	PetClinic_Demo...	PetClinic_Demo...	Passed		10	2016/02/09 16:37
10	Run_2-9_7.3...	PetClinic_Demo...	PetClinic_Demo...	Failed		14	2016/02/09 16:34
9	Run_1-27_9...	PetClinic_Demo...	PetClinic_Demo...	Passed		11	2016/01/27 18:48
8	Run_1-27_9...	PetClinic_Demo...	PetClinic_Demo...	Passed		12	2016/01/27 18:46

A detailed view of the failed test run (ID: 12) is shown in a pop-up window. It displays the test name 'Run_2-9_10...', type 'QUICKTEST_TEST', and a table of defects. The defect table has columns for '不具合ID' (Defect ID), '不具合: サマリ' (Defect: Summary), 'リンクされた工...' (Linked Work...), '被リンクステータス' (Linked Status), and 'リンクのコメント' (Link Comment).

不具合ID	不具合: サマリ	リンクされた工...	被リンクステータス	リンクのコメント
2	Checkpoint failed...	Run <Run_2-...	Failed	

The screenshot shows the HP Application Lifecycle Management interface displaying a defect detail view. The top navigation bar includes '不具合(D)' (Defects), '編集(E)' (Edit), '表示(V)' (View), 'お気に入り(F)' (Favorites), and 'アナリシス(A)' (Analysis). The main area displays a table of defects with columns for 'コメント' (Comment), 'サブジェクト' (Subject), 'サマリ' (Summary), 'ステータス' (Status), 'ターゲットサイクル' (Target Cycle), 'ターゲットリリース' (Target Release), 'プロジェクト' (Project), and '起因するコード...' (Causing Code...).

コメント	サブジェクト	サマリ	ステータス	ターゲットサイクル	ターゲットリリース	プロジェクト	起因するコード...
		Checkpoint failed...	新規				
		Checkpoint failed...	新規				

The defect detail view shows the following information:

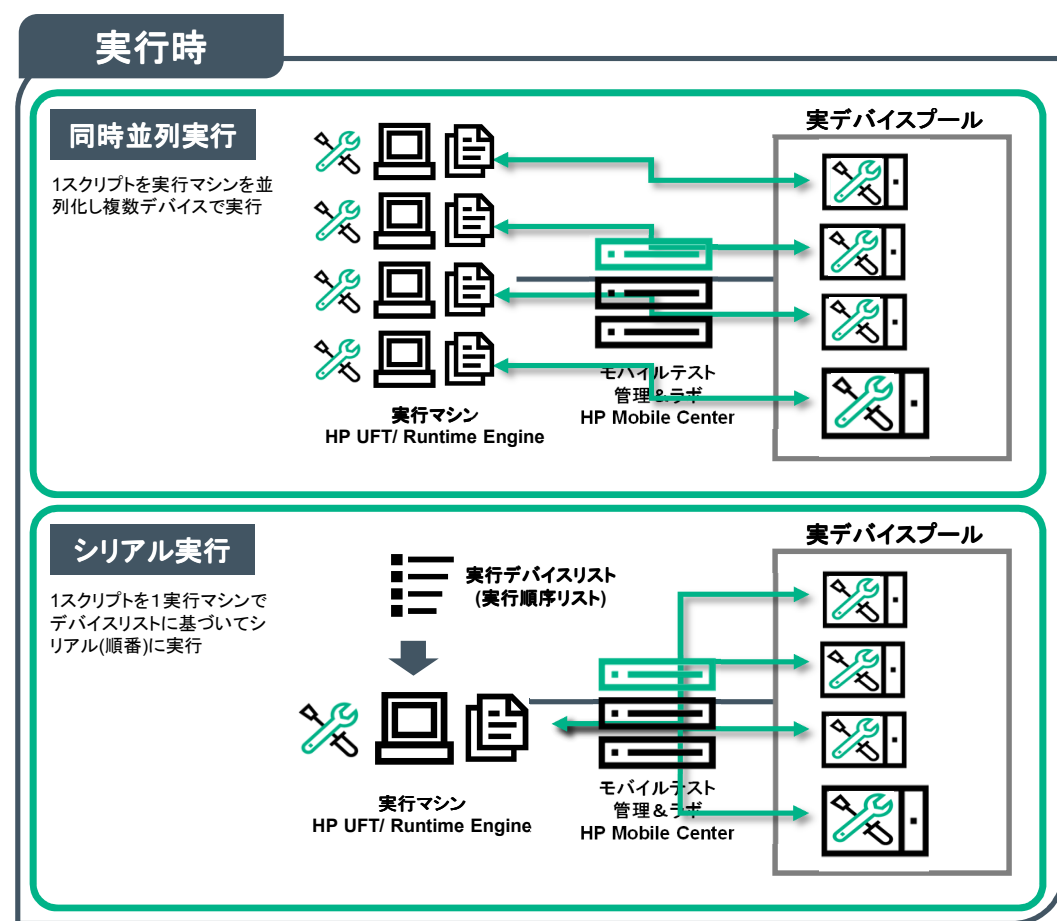
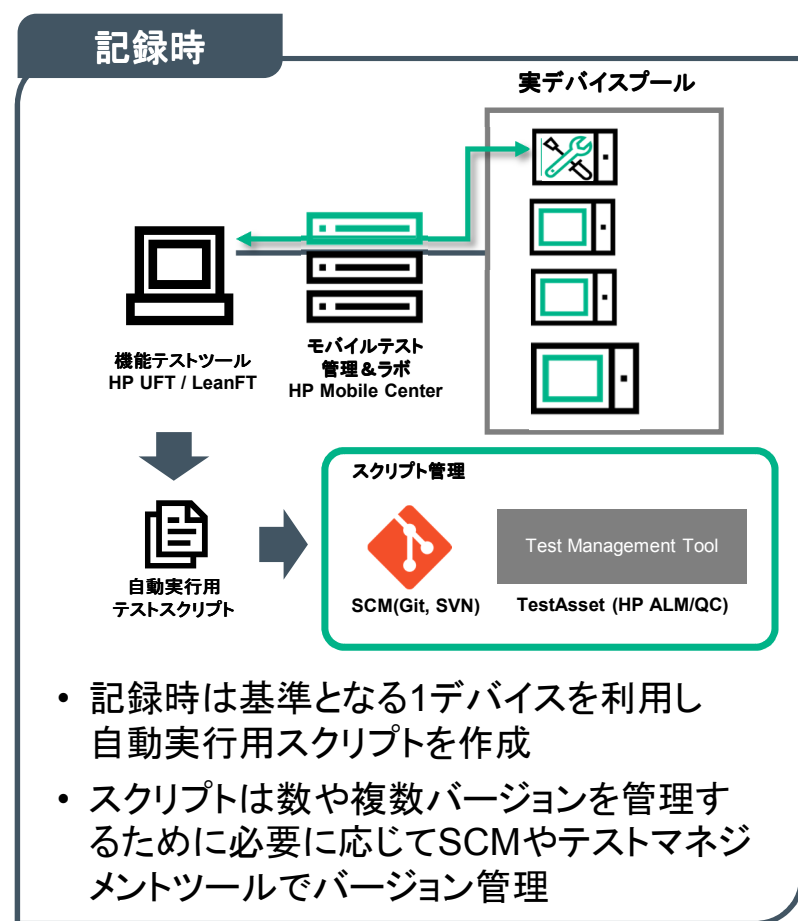
- サマリ:** Checkpoint failed in HP Unified Functional Testing
- 説明:** This defect was added automatically by HP Unified Functional Testing
- ビットマップ:** チェックポイント "pets" failed
- Test name:** PetClinic_Demo_01
- Test location:** Subject#PetClinic#PetClinic_Demo_01
- Action name:** Action 1
- Operating system:** Windows 7
- Host:** EVAN

A chart titled 'テスト進行状況 - ステータスによるグループ化' (Test Progress Status - Grouped by Status) is shown. The Y-axis represents the number of tests (テストの数) from 0 to 30. The X-axis represents time (時間) from 10-9月 to 11-7月. The legend indicates the following categories: Blocked (blue), Failed (red), No Run (orange), Not Completed (purple), and Passed (green).

プロセス連携を踏まえた モバイルテスト自動化

1. 自動化取り組みの前に知っておく事と注意点
2. 自動化率について知るべきポイント
3. HPE UFT & HPE MC を使った自動化スクリプト作成手順
4. テスト用モバイルデバイス & アプリ管理(ラボ管理)
5. CI/CD自動化プロセスへの統合方法とTIPS
6. 複数デバイス & アプリケーションのコントロール方法
7. インフラ構築 & デプロイプロセスとテストツールの連携方法

1つのスクリプトを複数デバイスで実行し効率化(注)



自動スクリプトによるマルチデバイス、アプリコントロール記述方法

サンプルスクリプト



```
Device("Device_1").App("MyApp").Launch DefaultInstallBehavior, DefaultRestartBehavior  
Device("Device_1").App("MyApp").MobileList( tabItemFirst ).Select 5,0  
Device("Device_1").App("MyApp").MobileDropDown("RegularPicker").Select "Arman",0  
...  
Device("Device_2").App("AnotherApp").Launch DefaultInstallBehavior, DefaultRestartBehavior
```

オブジェクトプロパティのデバイスIDをパラメータ化

パラメータをデータテーブルでデータ駆動適用

	id1	id2	Device ID	D	E	F	G
1	0a9cdee3	0cr3cxp2					
2	0cr3cxp2	0a9cdee3					
3	0cr3cxp2	72eTas88					
4	0a9cdee3	72eTas88					
5	0a9cdee3	0cr3cxp2					
6	0cr3cxp2	0a9cdee3					
7	0cr3cxp2	72eTas88					
8	0a9cdee3	72eTas88					

Launch関数引数 (object.Launch [InstallApp], [RestartApp])

- InstallApp
 - DefaultInstallBehavior: 「Record and Run Settings」設定に基づきアプリをインストール
 - DoNotInstall: インストールしない
 - Install: 「Record and Run Settings」設定を無視してインストール
- RestartApp
 - DefaultRestartBehavior: 「Record and Run Settings」設定に基づきアプリを再起動
 - DoNotRestart: 再起動しない
 - Restart: 「Record and Run Settings」設定を無視して再起動

プロセス連携を踏まえた モバイルテスト自動化

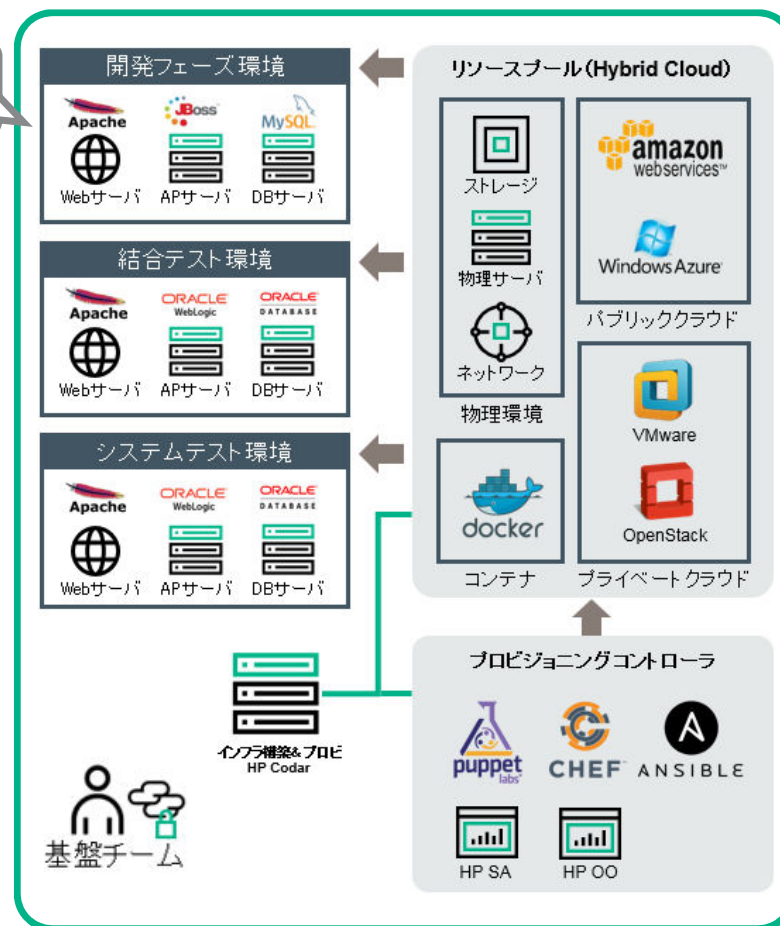
1. 自動化取り組みの前に知っておく事と注意点
2. 自動化率について知るべきポイント
3. HPE UFT & HPE MC を使った自動化スクリプト作成手順
4. テスト用モバイルデバイス & アプリ管理(ラボ管理)
5. CI/CD自動化プロセスへの統合方法とTIPS
6. 複数デバイス & アプリケーションのコントロール方法
7. インフラ構築 & デプロイプロセスとテストツールの連携方法

インフラ環境で動的に変わる環境情報がある場合の課題と対応

リソースプールから環境を切り出した際にIPアドレスが変わる

- クラウド環境でオーケストレーションツールで構築された環境においてIPアドレスなど環境情報が動的に変化する場合、テストスクリプトが動作しない
- 回避策としては、次が考えられる(IPアドレスの場合)
 - 割り当てられたIPアドレスを取得し DNS を修正するスクリプトを実行する
 - 予めテストスクリプトのアドレス部分をパラメータ化しておき、割り当てられたIPアドレスを取得しスクリプトを実行する

次スライドで2の方法を説明します



テストツールのアクセス先を動的に変更する設定例(UFTパラメータ利用)

Codar

デプロイメントステータス

トポロジ

Application Server_1

ipAddress
192.168.3.124

プラットフォームモデルで定義された自動デプロイに基づき Codar で動的環境情報 (IPアドレス等) を取得可能

HPE OO などを利用し HPE UFT の実行時の動的パラメータとして渡す処理を定義しておく

プロビジョニングコントローラ

- puppet labs
- CHEF
- ANSIBLE
- HP SA
- HP OO

UFT

アクション呼び出しプロパティ

実行 パラメータの値

名前	タイプ	標準設定値	記述
URL_IP	String	192.168.3.107:8080	

外部からの動的パラメータを受け取り利用できるように入力パラメータを設定

出力パラメータ:

名前	タイプ	保管先:	記述
----	-----	------	----

入力パラメータを利用してURLを変更できるようにスクリプトを編集

```
1 '記録時の固定URL  
2 'Browser("PetClinic :: a Spring").Navigate "http://192.168.3.107:8080/petclinic/  
3  
4 'Codarからの動的URLを取得  
5 param1 = Parameter("URL_IP")  
6 url = "http://" & param1 & "/petclinic/"  
7 Browser("PetClinic :: a Spring").Navigate url
```

まとめ

HPE の優位性

Capgemini 社のWQRに7年連続で協力、企業の品質への取り組み状況をリサーチ

- 最新版(2015年10月版)では、世界の1500人以上の開発者、テスターを対象にリサーチを実施
- 2014年度に比べ、テストケースに対する自動化率が28%から45%まで大きく伸びており、世界的にもテスト自動化にフォーカスが当たっている
- キーベネフィットとして「早い不具合検出」、「テスト内容の透明性」、「テストサイクル時間の短縮」、「テスト費用の削減」、「再利用性」が認識されており、テスト自動化への投資を後押ししている



対象企業内テスト自動化率



まとめ

本セッションのまとめ

- モバイルアプリケーションの自動化は、デバイスで動作するアプリの機能テストを自動化するスクリプト作成をどうするかにフォーカスされがちだが、自動化の範囲としてどこまで本来はカバーすべきか、期待する自動化範囲に対する自動化率を押さえて取り組むのがよい。
- テスト自動化ツールをCI/CDプロセスの中で活用するためにはビルドしたアプリパッケージのテスト環境デバイスへのデプロイ、CIツールからのテスト実行連携、品質計測のためのテスト管理ツールとの連携が出来るかを確認しておくといよい。
- DevOpsを実現する中でインフラ構築&プロビツールによる動的環境情報がテストスクリプト実行に影響するのであれば、どのように動的環境情報を取得し利用できるかを事前に確認しておくといよい。

「どこから手をつけるべきか」や「進め方のベストプラクティス」などお気軽にご相談ください



Hewlett Packard
Enterprise

Thank you

Contact information