NTT DATA
Trusted Global Innovator

ロボットアームを用いた、
スマホアプリのテスト自動化の事例紹介

株式会社NTTデータ 公共・社会基盤事業推進部 プロジェクト推進統括部
中里 大貴
宮越 一稀

© 2023 NTT DATA Corporation

自己紹介

所属

株式会社NTTデータ
公共・社会基盤事業推進部　プロジェクト推進統括　技術戦略担当

経歴

中里　大貴

・インフラエンジニア(7年目)
・仮想化基盤構築やCICD環境構築に従事
　- 大手通信業者の仮想化基盤構築
　- 仮想化基盤の自動構築/自動テスト実施
　- スマートフォンアプリのCICD環境構築

宮越 一稀

・ソフトウェアエンジニア(3年目)
・アプリ開発や自動テスト環境構築に従事
　- スマートフォンアプリの開発（要件定義～設計）
　- スマートフォンアプリ開発における自動テスト環境構築

© 2022 NTT DATA Corporation

テーマとアジェンダ

テーマ

物理カード読み取り操作が必要なスマホアプリの開発において、テスト自動化を行った事例

お伝えしたい事

テスト自動化におけるロボットアーム適用のノウハウ・苦労点

アジェンダ

・テスト自動化を実行する仕組みについて

・自動化するにあたっての課題とその対応

・自動化の効果

・ロボットアーム適用における課題・苦労点・ノウハウ

© 2022 NTT DATA Corporation

取り組み概要

背景

スマートフォンアプリの開発の際に多端末テストを定期的に行う方針だが、
対象が多く人手で実施するとなると多大なコストが継続的に発生する

⇒**テスト自動化の仕組みを構築**

対象のテスト

アプリ対象全機種に対する品質担保のため、機種特性に起因するバグがないか検証する記述式テスト

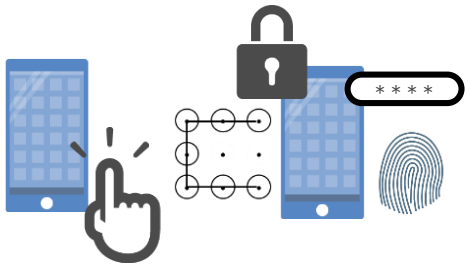
ポイント①

スマホの基本操作に加え、**物理操作**を必要とする
→**エミュレータが適用不可**

実施するアプリ操作

スマホの基本操作
- タップ
- 文字入力
- スクロール
- 認証操作

物理操作
- **カードの読取**

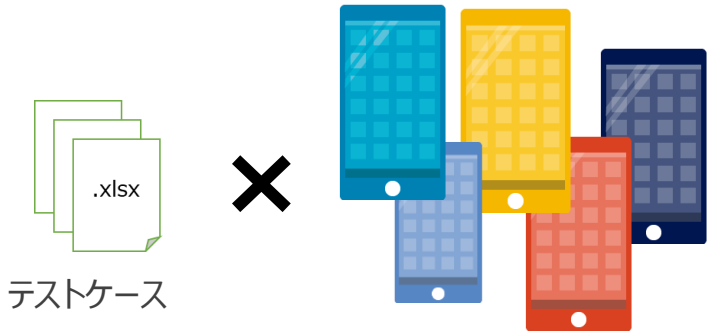


ポイント②

全機種をテスト対象とするため、莫大な工数がかかる

テスト対象

機種：Android **全91機種**
OS　：Android 9～13



© 2022 NTT DATA Corporation

課題①とその対策　～ロボットアームを利用したカード読取操作の自動化～

課題

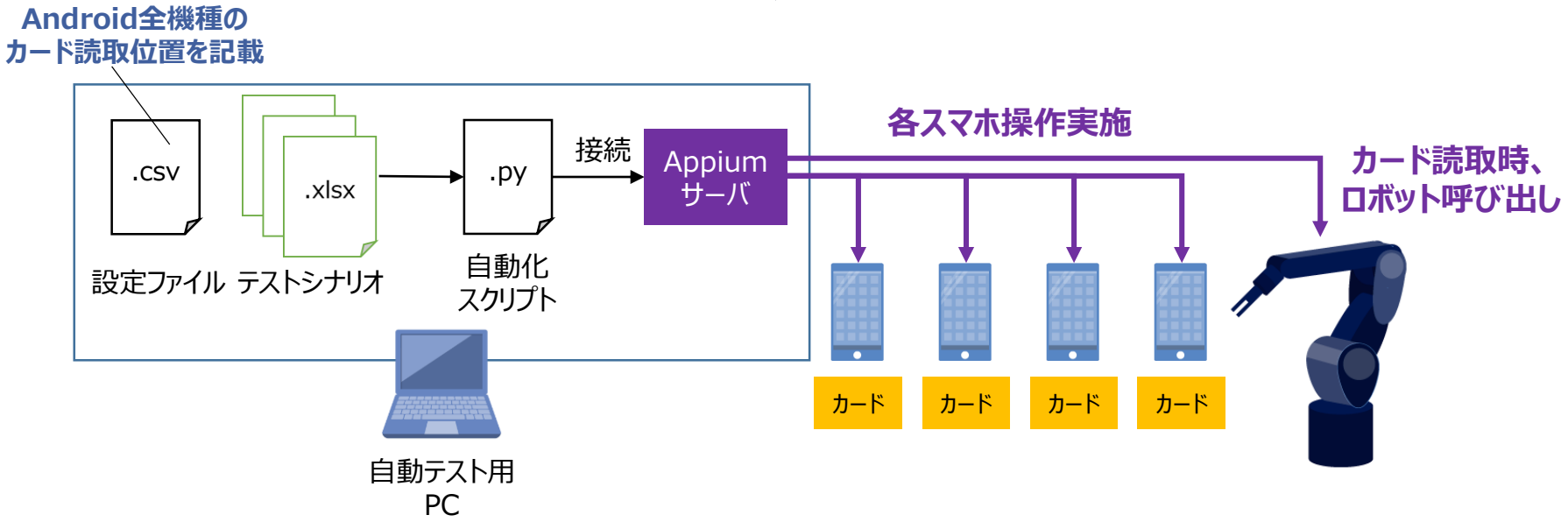
従来のスマホの画面操作を自動化するツール
⇒スマホの物理操作が想定されていない

物理カードを読み取る仕組みの構築が必要

対策

物理カード読取操作について、一連の操作を自動化
- **ロボットアームを利用**
- スマホ操作(Appium) とロボットアームを連携

Android機種毎にカード読取位置が異なるため、
事前に位置座標を測定し、設定ファイルに記載

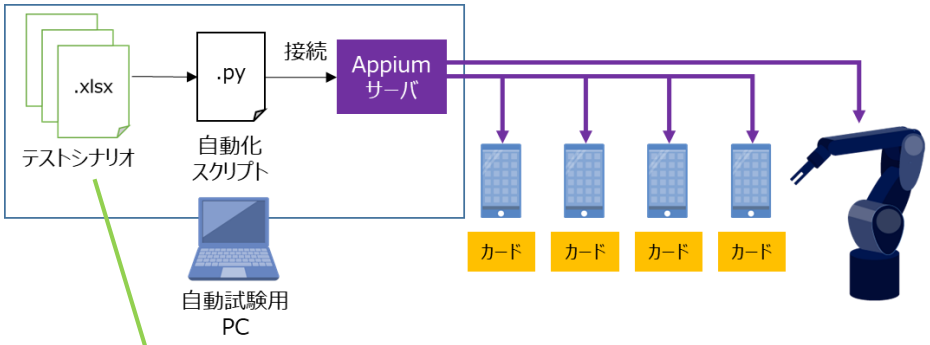


© 2022 NTT DATA Corporation

課題①とその対策　～ロボットアームを利用したカード読取操作の自動化～

特徴

テストシナリオはExcelで作成し、各動作はキーワードで指定可能となるように実装
⇒**キーワード駆動テストの実行**を可能としている。
キーワード駆動テスト：操作を表す**「キーワード」**と、対象となる**「データ」**で構成された表を使って行うテスト

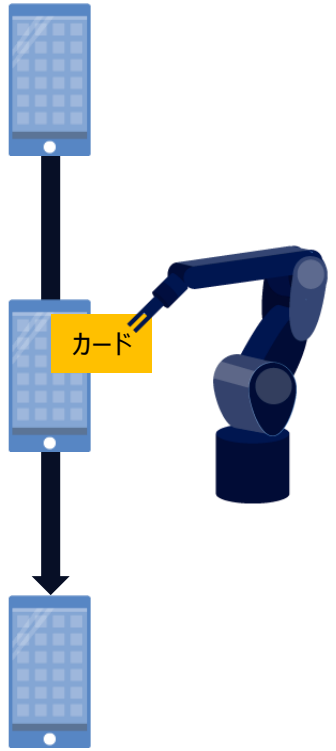


手順	画面名	要素名	動作	入力値
1	トップ画面	メニューボタン	タップ	-
2	メニュー画面	カード登録ボタン	タップ	-
3	文字入力画面	入力フォーム	入力	123456
4	文字入力画面	次へボタン	タップ	-
5	カード読取画面	-	**ロボット操作**	-
6	読取完了画面	次へボタン	タップ	-

データ　　**キーワード**

スマホ操作

ロボットによる
カード読取操作



スマホ操作

© 2022 NTT DATA Corporation

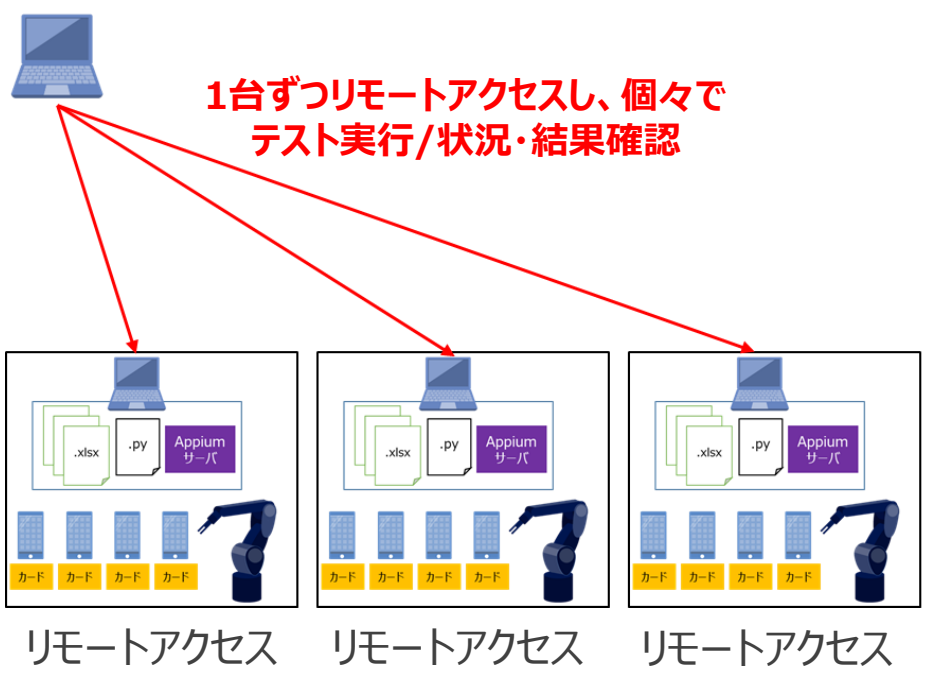
課題②とその対策～GitLab CIを利用した全テスト実行環境の並列化～

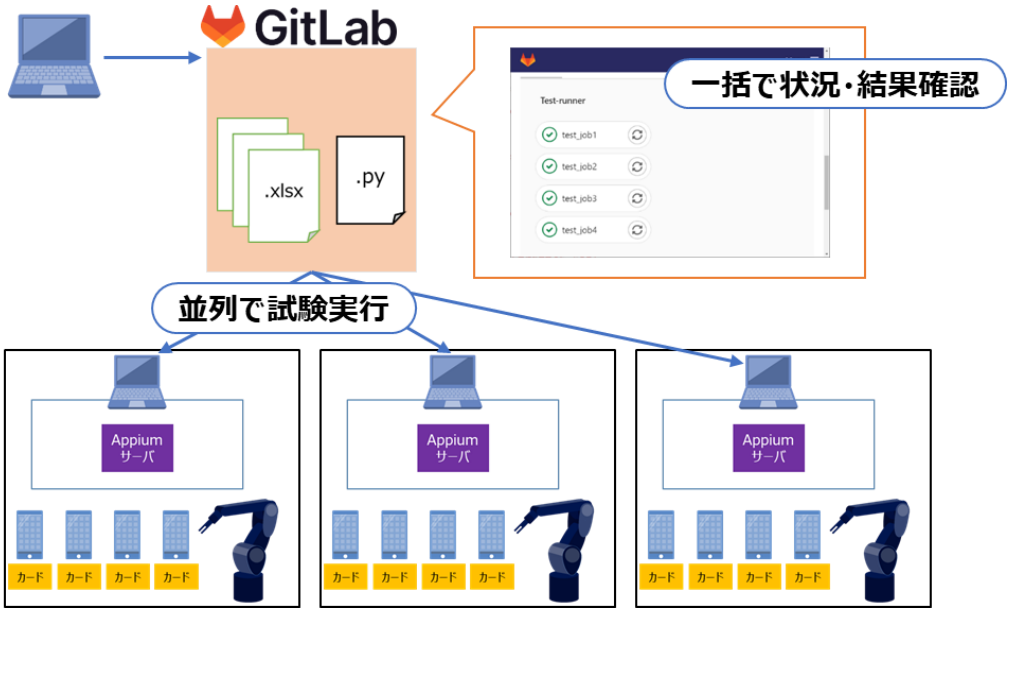
課題

テスト対象の端末数が多く、テスト実行結果の
確認や資材インストールの際の手間が多い

対策

GitLab CIを利用し、全体処理を改善
- 各テスト実行環境へのリモートアクセス不要
- 並列でテスト実行/結果確認/資材インストール





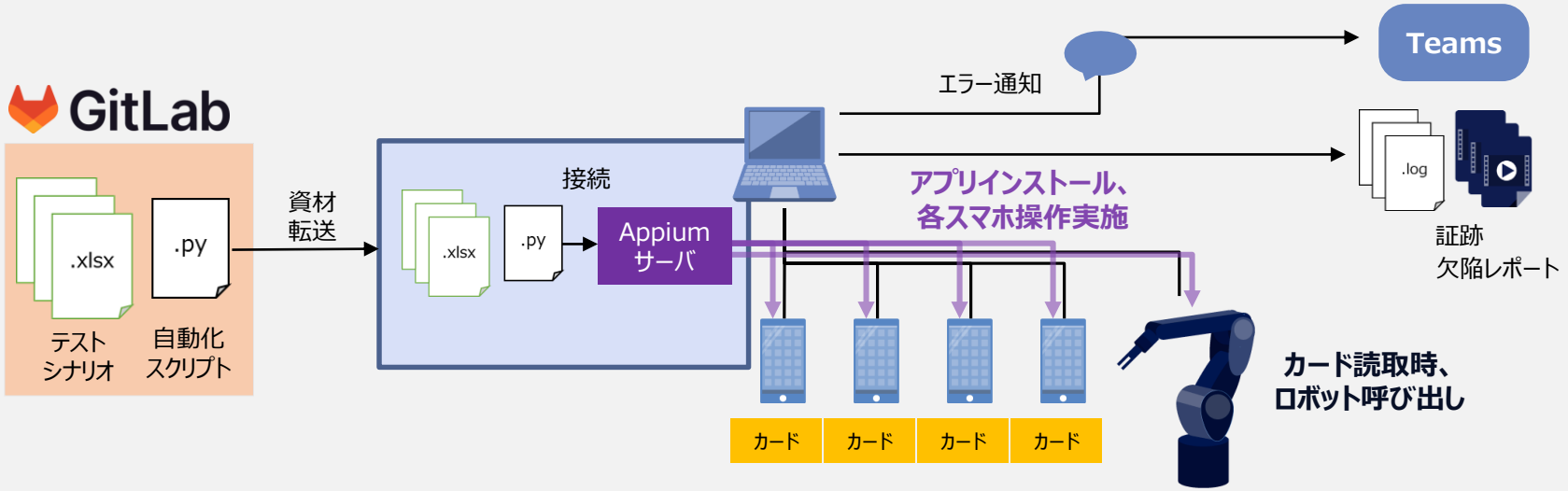
テスト実行環境へ1台ずつリモートアクセスが必要

全テスト実行環境へまとめてアクセス可能

© 2022 NTT DATA Corporation

テスト自動化の仕組み

全体像



使用したツールなど

言語　　：Python　　・・・　ロボットアームの操作やAppiumスクリプトで使用
リポジトリ　：Gitlab　　・・・　自動化スクリプトやテストシナリオを格納
　　テスト実行を各端末に指示する(Gitlab CICD)
テスト自動化：Appium　　・・・　スマホアプリ自動化ツール
物理操作　　：ロボットアーム・・・「DOBOT Magician®」を使用

© 2022 NTT DATA Corporation

参考　テスト自動化ツールの選定

対象のスマホアプリの特徴から、「**画面要素認識方式**」を選定。さらに、画面要素認識方式のツール群の中から「**Appium**」を選定。

■今回の対象スマホアプリの特徴
・オブジェクトの位置は機種/OSに依存
・オブジェクトのデザインが頻繁に変更

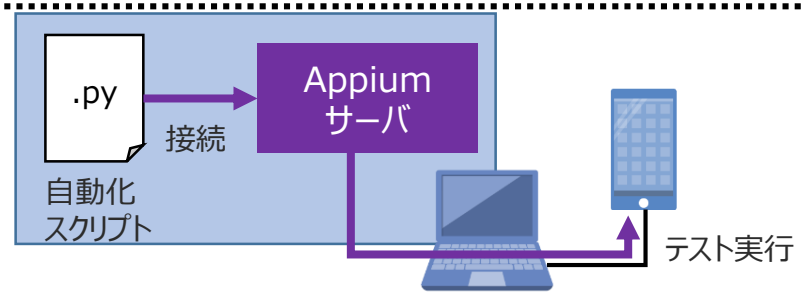
方式	概要
座標指定方式	画面上の位置情報（x,y）に基づき、対象オブジェクトを操作
画像認識方式	マッチング画像を用いて対象画面を検索し、一致したオブジェクトを操作
画面要素認識方式	対象画面の画面要素のプロパティ情報（要素ID等）に基づき、オブジェクトを操作

参考：JaSST'20 Hokkaido
「SikuliXを用いた リグレッションテスト自動化の事例紹介」(NTTデータ)

画面要素認識方式のツール	ツール選定における要件		
	①スマホアプリ自体の改修不要か	②OSバージョンアップに追従しているか	③マルチプラットフォームに対応しているか
Appium	○	○	○
Espresso	×	○	△
Calabash	△	×	○
EarlGray	×	○	△
他6ツール			

Appium

OpenJS Foundation提供の、
モバイルアプリ向けテスト自動化ツール



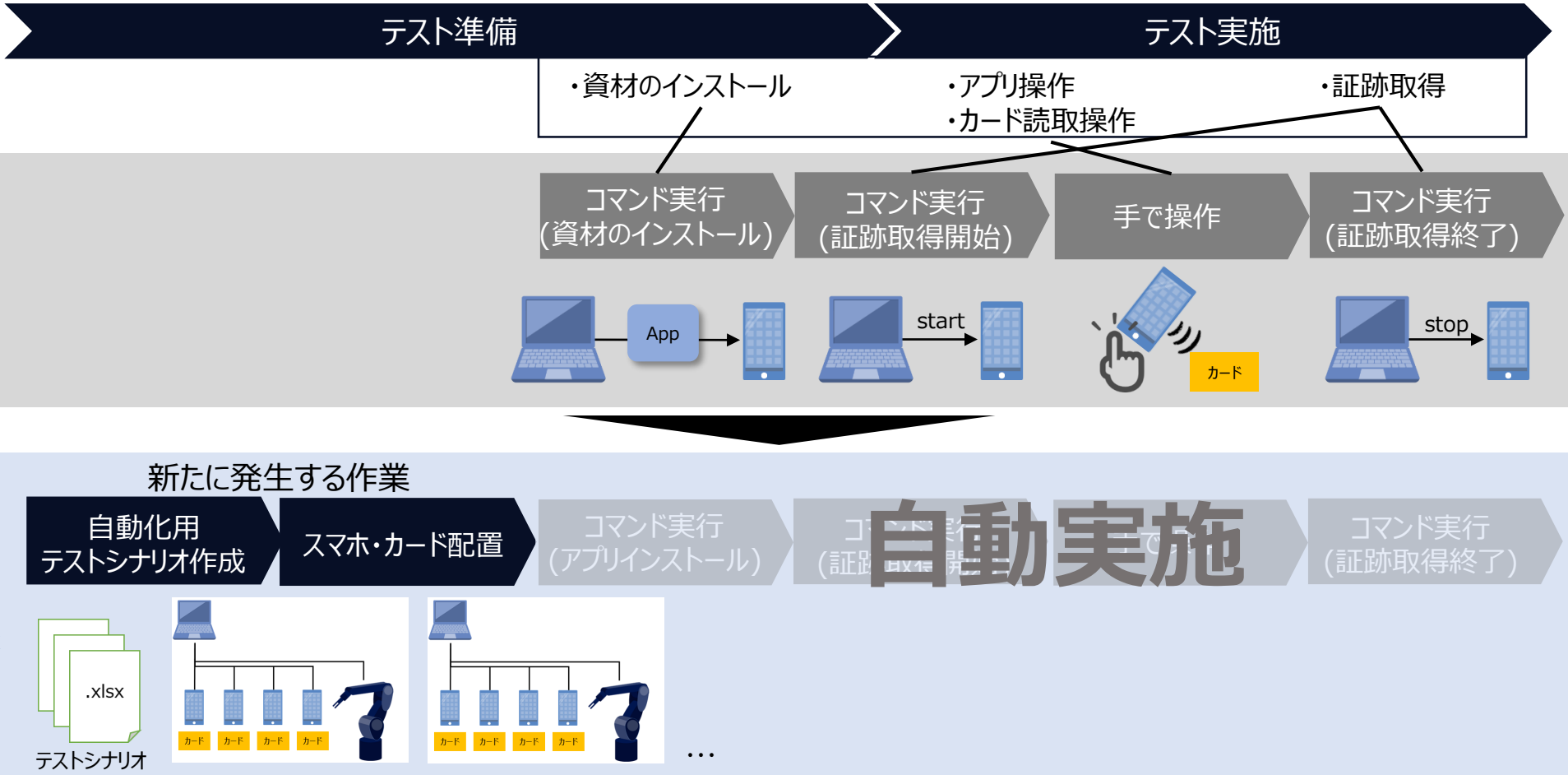
Appiumのテスト実行イメージ

© 2022 NTT DATA Corporation

効果

効果（作業プロセス）

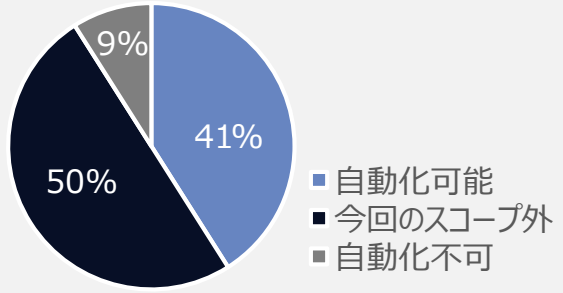
本自動化ツールにより、**「テスト実施」における全作業を自動化**



効果

効果（自動化率）

・全テストケースの約41%自動化



今回スコープ外としたテストケース

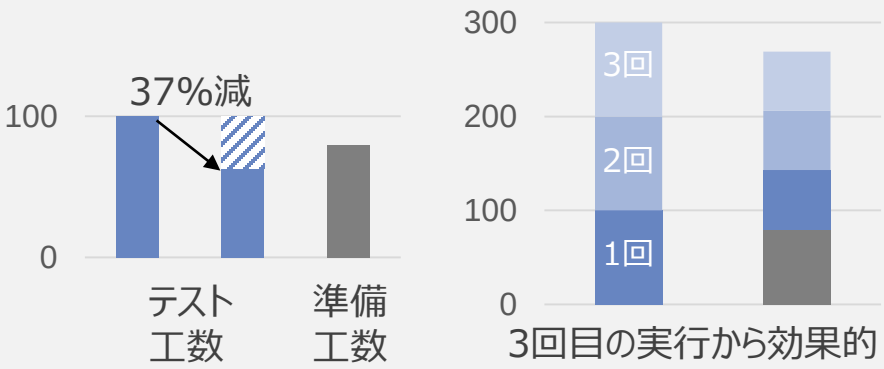
・1テストケースにスマホを複数台必要とするもの
・他アプリ操作/ブラウザ操作を必要とするもの
・スマホのカメラ読取を必要とするもの
→ネクストステップとして検討予定

自動化不可のテストケース

・生体認証を必要とするケース

効果（工数）

・テスト工数は計約37%の削減
・物品の費用や準備工数で約80%の工数がかかっている
・今後の運用にて同様のテストを3回実施することで、
効果が上回る見込み

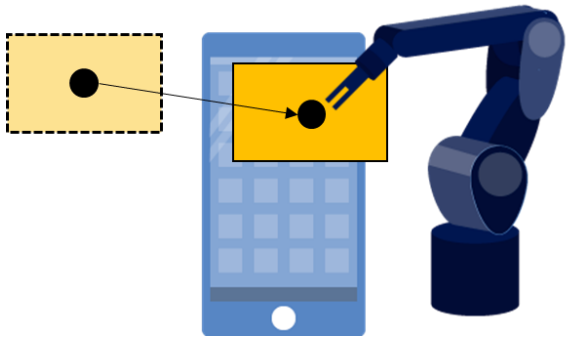


© 2022 NTT DATA Corporation

ロボットアームにおける工夫点①　Android機種によってカードの読取性能に差異

課題

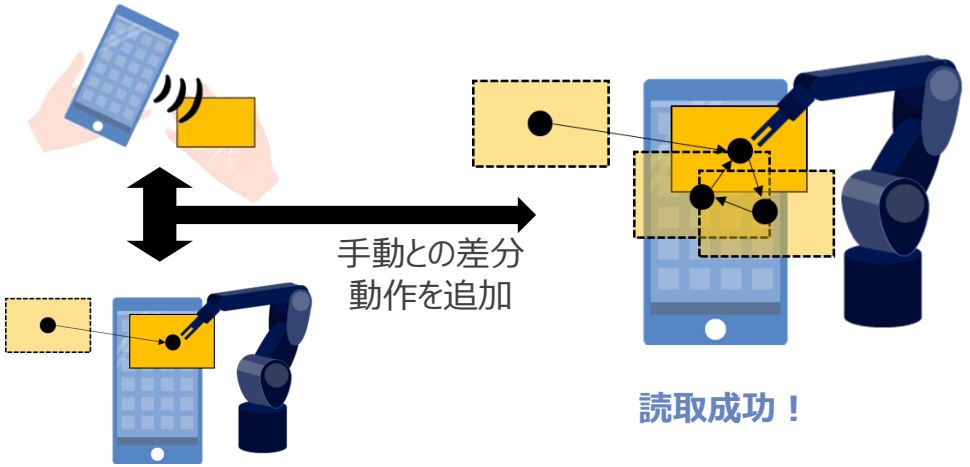
Androidの読取位置の1点にカードをかざすだけでは
読取可能な機種・不可能な機種が混在
↓
カード読取ミスによる**テストNG頻出**



一部の機種で
読取反応なし

対策

手動でかざす場合とロボットアームの動作を比較
↓
手振れ（微小なスライド動作）が必要と仮説立て
↓
1点にかざす＋差分動作（手振れ）を追加実装

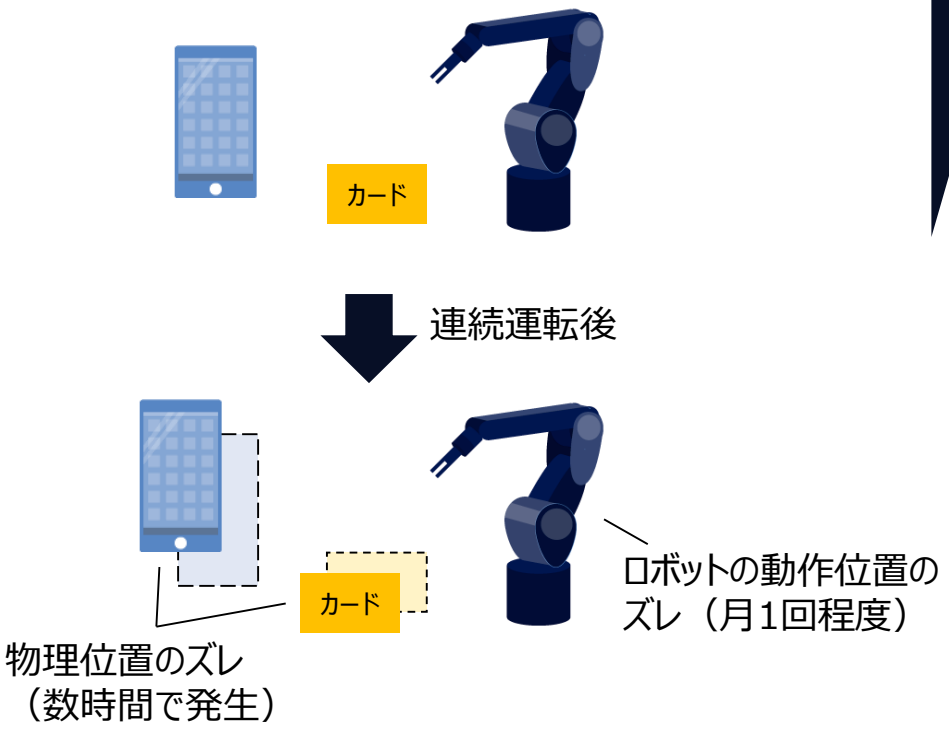


© 2022 NTT DATA Corporation

ロボットアームにおける工夫点②　連続運転時の位置ずれ

課題

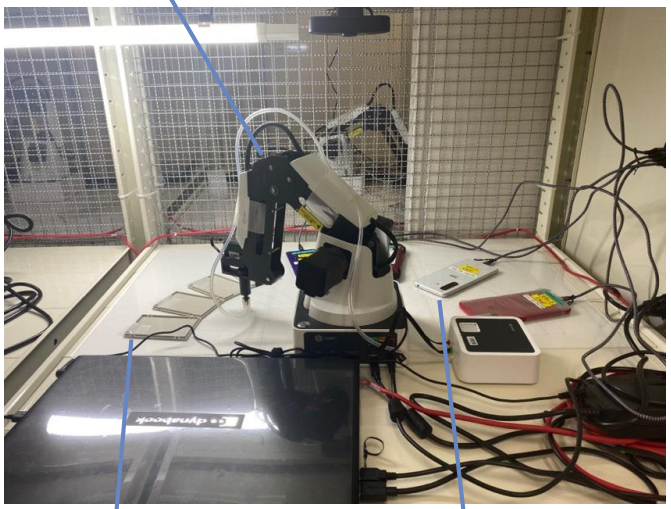
連続して運転すると
物理機器について徐々に位置ずれが発生する
↓
カードの**読み取りができなくなり**テストでNGとなる



対策

位置ずれが発生する箇所について環境/運用で対応

ロボットアーム:定期的な初期位置の調整（月1回）



カード:カードケースを配置

スマホ:本体を固定

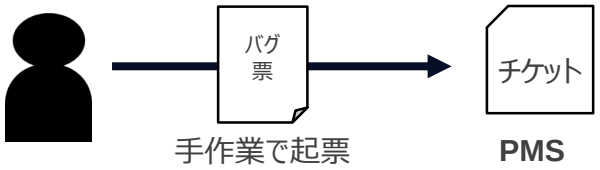
© 2022 NTT DATA Corporation

その他工夫点①　バグ票起票の半自動化

テスト準備	テスト実施	結果確認
・資材のインストール	・アプリ操作 ・カード読取操作 ・証跡取得	・テスト結果の確認 ・証跡の確認 ・バグ票の起票（エラー時）
初期実装		

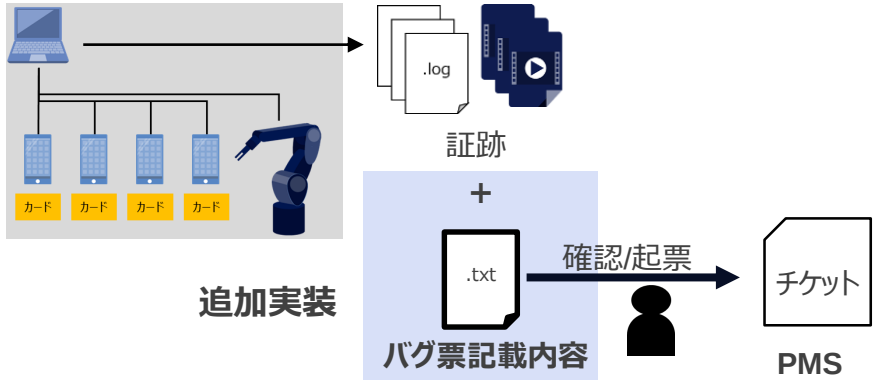
課題

バグ票に記載する情報が多く、
手作業で実施すると時間がかかる



対策

バグ票記載内容の自動出力機能追加実装
→バグ票記載作業不要に！

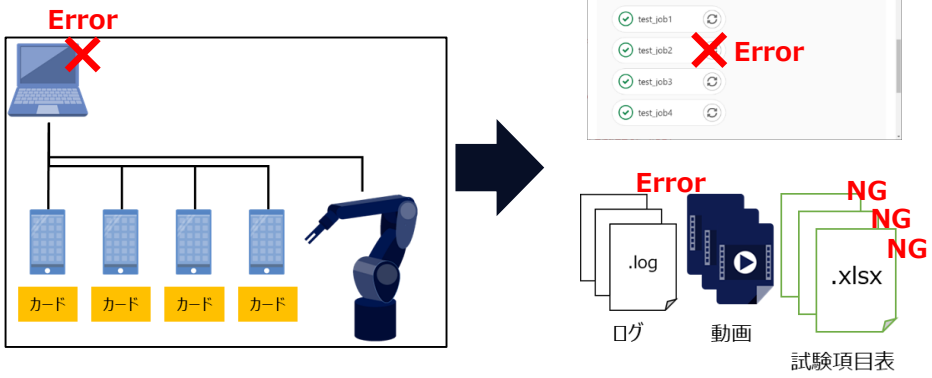


© 2022 NTT DATA Corporation

その他工夫点②　エラー発生時の通知

課題

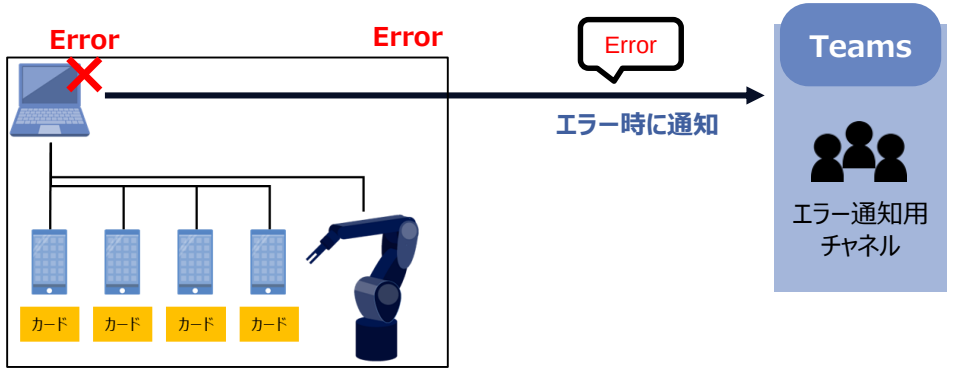
エラー発生により途中でテストが止まった際、
事象に気づくのに時間がかかっていた
↓
すぐに気づくためには、**実施状況をこまめに確認**する
必要があり、その分他作業にも影響が出た



証跡・実施状況を見るまで気づけない

対策

エラー発生時の**通知機能**を追加実装
↓
Teamsに通知することで、即座に気づける



© 2022 NTT DATA Corporation

得られたノウハウ

・ロボットアームを使ったテスト自動化には試行錯誤、物理的な工夫が必要

参考となる情報源が少ないため、自分で動かしてみるしかない。
物理的な課題は物理的な方法で解決するケースが多い（デバイスの位置ずれなど）。

・実機での多端末テストを自動化する際は、機種ごとに物理的にも特性確認が必要

同メーカー、同OSでも特徴や挙動が異なる場合がある。
（カードのかざす位置、読取性能、PC接続状態だと読み取り不可など）

・ロボットアームの利用には防音対策が必要

アームの移動時、カード持ち上げ時の音が非常に大きい。
ロボットアーム用に部屋を設ける、夜間帯にテスト実施するなどの対策が必要。

© 2022 NTT DATA Corporation

まとめと展望

まとめ

・テスト自動化を実行する仕組みについて
- ☑ 多端末に実行する記述式テストの自動化

・自動化するにあたっての課題とその対応
- ☑ ロボットアームで物理操作の自動化
- ☑ Gitlab×Appiumでテスト実行を並列化

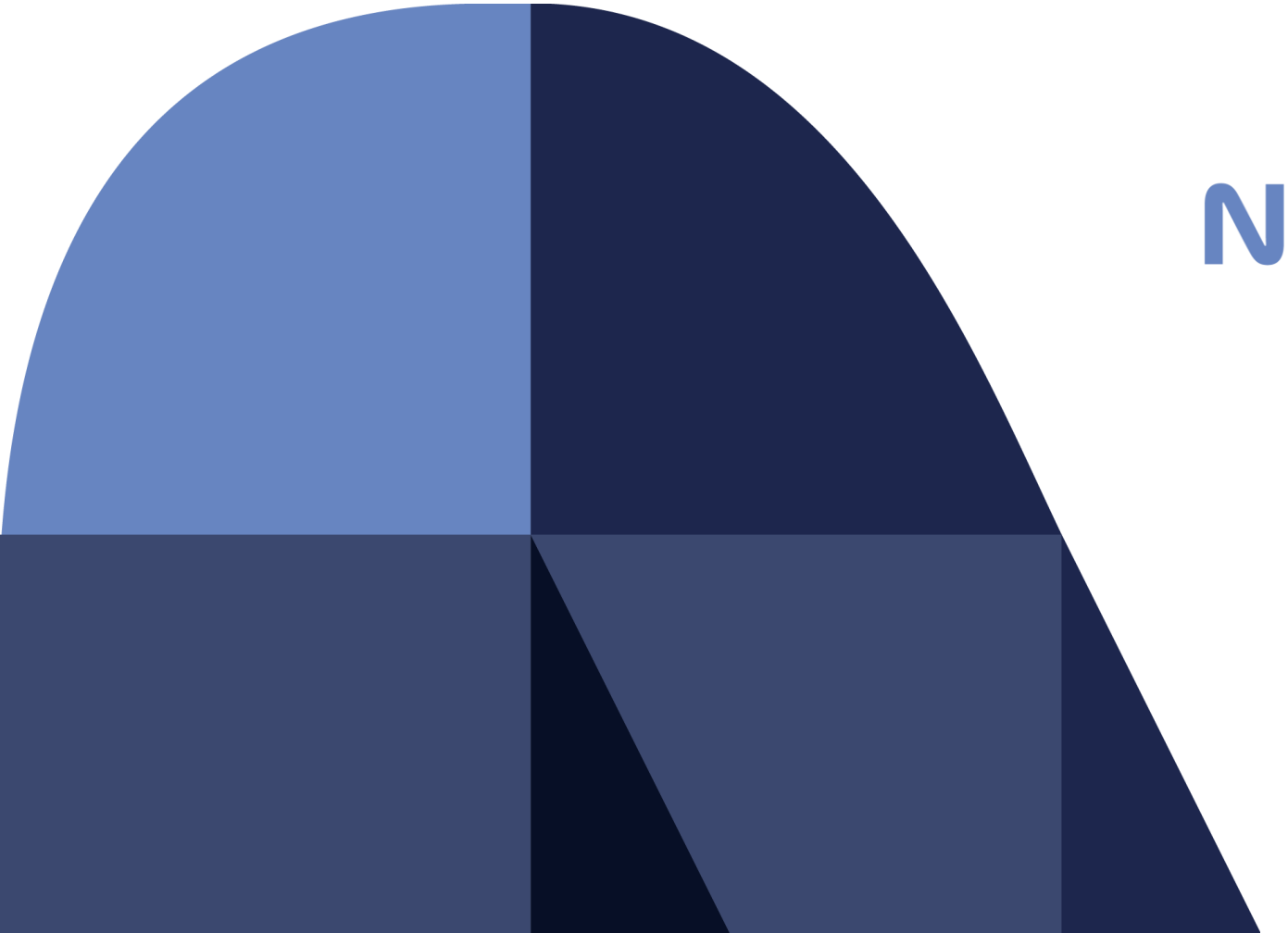
・自動化の効果
- ☑ 41%のテストを自動化し、37%の工数削減

・その他　課題・苦労点
- ☑ エラー発生時のバグ票起票の半自動化
- ☑ 連続運転時の位置ずれが発生しないように工夫
- ☑ スマホ機種のカード読取性能に影響を受けないよう、ロボットアームで手振れを実装
- ☑ エラー発生時にすぐ気がつけるように通知を行う

展望

今後は以下のような対応を行うことにより、自動化のスコープ拡大/汎用性向上を目指していく
- ☑ ロボットアームの他の用途の検証
  - 今回使用したのはカード読取操作のみ
  - その他考えられる用途：スマホカメラへのQRコードの投影、スマホのかざし動作など
- ☑ 画面比較による結果確認を自動化

© 2022 NTT DATA Corporation



NTT DATA

Trusted Global Innovator

© 2022 NTT DATA Corporation

参考　使用したロボットアームについて

DOBOT社提供のロボットアームを使用

特徴１．吸引機能

吸盤で吸引することで、カードの持ち運びが可能

特徴２．座標指定で動作

位置座標を指定することで、アームを操作可能

特徴３．プログラムから呼び出し可能

DOBOT社からロボットアーム操作用のAPI（Dobot API）が提供されており、プログラムから呼び出しが可能

https://techshare.co.jp/product/dobot/magician/

© 2022 NTT DATA Corporation