UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

MSSBOX, UM PACOTE PYTHON PARA FACILITAR A AUTOMAÇÃO DE TESTES MOBILE QUE ENVOLVEM SIM CARDS USANDO O HARDWARE MATRIX SIM SWITCH BOX

Relatório Técnico relativo ao Trabalho de Conclusão Curso do Bacharelado em Sistemas de Informação na modalidade Empresa

Aluno
Antonio Carlos Da Silva

Orientador
Silvana Bocanegra
DEINFO/UFRPE-Sede

Co-orientador
Renan Marques Gomes de Oliveira
CESAR School

03 de junho de 2022

Antonio Carlos da Silva

mssbox, um pacote python para facilitar a automação de testes mobile que envolvem sim cards usando o hardware Matrix SIM Switch Box

Relatório Técnico apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE
Departamento de Estatística e Informática
Curso de Bacharelado em Sistemas de Informação

Orientador: Silvana Bocanegra
Co-orientador: Renan Marques Gomes de Oliveira

Recife
Junho de 2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

S586m Silva, Antonio Carlos Da
Mssbox, um pacote python para facilitar a automação de testes mobile que envolvem sim cards usando o hardware matrix SIM switch box / Antonio Carlos Da Silva. - 2022.
23 f. : il.

Orientadora: Silvana Bocanegra.
Coorientador: Renan Marques Gomes de Oliveira.
Inclui referências.

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, Bacharelado em Sistemas da Informação, Recife, 2022.

1. Matrix SIM switch box. 2. Testes de software. 3. Automação de testes. 4. Testes Mobile. 5. MSSB. I. Bocanegra, Silvana, orient. II. Oliveira, Renan Marques Gomes de, coorient. III. Título

CDD 004

ANTONIO CARLOS DA SILVA

MSSBOX, UM PACOTE PYTHON PARA FACILITAR A AUTOMAÇÃO DE TESTES MOBILE QUE ENVOLVEM SIM CARDS USANDO O HARDWARE SIM SWITCH BOX

Artigo apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Aprovada em: 3 de Junho de 2022.

BANCA EXAMINADORA

Silvana Bocanegra (Orientador)
Departamento de Estatísitca e Informática
Universidade Federal Rural de Pernambuco

Rodrigo Elia Assad
Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco

Guilherme Vilar
Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco

Resumo. *Os aplicativos móveis requerem testes como qualquer outro sistema de software. Entretanto executar testes móveis é bastante desafiador, pois é necessário pensar cuidadosamente em cada tipo de teste quando se pensa em criar casos de teste. Para automação, os desafios são ainda maiores devido a variedade de dispositivos, sistemas operacionais e questões de redes móveis. Trazendo para o cenário de testes em aparelhos celulares, uma das formas de testar considerando a rede móvel é inserindo um cartão SIM no aparelho. A necessidade de se manipular esses cartões SIM durante os testes é o desafio que esse trabalho busca explorar. O presente trabalho tem como objetivo principal apresentar uma ferramenta para auxílio na automação dos testes que necessitam dessa manipulação e o desenvolvimento de um pacote que permite utilizar a ferramenta na automação. Para isso, foi realizada uma pesquisa-ação através de um experimento em um projeto de testes móveis, no qual foi possível descrever a ferramenta e suas funcionalidades, bem como os passos necessários para sua utilização. Também é descrito como o pacote criado foi estruturado e os testes implementados para utilização do pacote. Por fim, foi possível identificar os benefícios que podem ser obtidos utilizando essa ferramenta para execução e automação de testes em aparelhos celulares.*

Palavras-chave: Matrix SIM Switch Box. Teste de Software. Automação de Testes. Testes Móveis. MSSB.

Abstract. *Mobile apps require testing like any other software system. However, running mobile tests is quite challenging, as you need to think carefully about each test type when thinking about creating test cases. For automation, the challenges are even greater due to the variety of devices, operating systems and mobile network issues. Bringing it to the testing scenario on cell phones, one of the ways to test considering the mobile network is by inserting a SIM card in the device. The need to manipulate these SIM cards during testing is the challenge that this work seeks to explore. The present work has as main objective to present a tool to aid in the automation of tests that need this manipulation and the development of a package that allows the tool to be used in automation. For this, an action research was carried out through an experiment in a mobile testing project, in which it was possible to describe the tool and its functionalities, as well as the necessary steps for its use. It is also described how the created package was structured and the tests implemented to use the package. Finally, it was possible to identify the benefits that can be obtained using this tool for test execution and automation on cell phones.*

Keywords: Matrix SIM Switch Box. Software Testing. Testing Automation. Mobile Testing. MSSB.

Sumário

1. Introdução

Diante da transformação da era digital, muitas tecnologias emergiram com o objetivo de trazer praticidade, velocidade e otimização do tempo para o dia a dia das pessoas e empresas. Nesse sentido, "mobile" significa muito mais do que celular. "Mobile" tornou-se uma cultura que está presente em grande parte das atividades cotidianas [Carvalho 2019]. A tecnologia mobile, em português, tecnologia móvel é uma categoria que consiste em uma ampla gama de dispositivos, com novas tecnologias bidirecionais sendo criadas a cada dia com usos exclusivos. Independentemente do formato, todos eles são vinculados por sua capacidade de enviar e receber sinais por meio da comunicação com outros dispositivos em redes [In 2021].

A IBM [IBM 2021] define tecnologia móvel como a tecnologia que vai onde o usuário vai. Ele consiste em dispositivos portáteis de comunicação bidirecional, dispositivos de computação e a tecnologia de rede que os conecta. Atualmente, a tecnologia móvel é caracterizada por dispositivos habilitados para internet, como smartphones, tablets e relógios. Estes são os últimos em uma progressão que inclui pagers bidirecionais, notebooks, telefones celulares, dispositivos de navegação GPS e muito mais.

Destacando os telefones celulares, chamados hoje de smartphones, a tecnologia envolvida nesses aparelhos impressiona, pois facilitam o nosso dia a dia através de suas funcionalidades e possibilidades. Um exemplo de como os smartphones estão inseridos na nossa sociedade está na alta quantidade de aparelhos comercializados somente no ano de 2020, ano em que foram vendidos 1.38 bilhões de celulares. Em 2018, 38% da população mundial possuía um smartphone, número que subiu para 46,5% em 2020. Só nos Estados Unidos, a venda de smartphone foi planejada em cerca de 77.5 bilhões de dólares americanos em 2019, um aumento de 18 bilhões em relação ao ano de 2010 [O'Dea 2021].

Almafitano [Amalfitano et al. 2013] pontua que muitas características dos dispositivos portáteis influenciam as estratégias de desenvolvimento e teste de aplicativos móveis. O autor discorre que duas características principais são a heterogeneidade das configurações de hardware dos dispositivos móveis e a variabilidade de suas condições de funcionamento. Dispositivos móveis comerciais vêm equipados com diferentes tipos e números de sensores de hardware, vários tipos de telas, CPUs com diferentes características, e assim por diante. Além disso, os dispositivos suportam várias redes móveis e tecnologias de comunicação.

Diante desse contexto, é necessário realizar alguns testes que envolvem as redes móveis, e para isso, é preciso executar o teste com determinado cartão SIM inserido no aparelho. Para vista de automação, gerenciar esses cartões SIM específicos é um desafio para que sejam criados testes automatizados, pois alguns testes necessitam que o cartão SIM seja inserido ou removido do aparelho. Assim, a problemática apontada por este trabalho se mostra pela dificuldade de se automatizar testes em dispositivos móveis quando é necessário manipular os cartões SIM durante o teste. Esse problema é encarado em um projeto de testes móveis da empresa CESAR, o qual foi o alvo do experimento.

Partindo desse cenário, pode-se questionar se esses sistemas e softwares incluídos nos smartphones são testados de forma eficiente. E mais, a automação já é utilizada no projeto? Como automatizar testes móveis em que é necessário inserir ou remover um cartão SIM do dispositivo? Este trabalho busca desenvolver uma solução para ajudar a

responder essas perguntas.

O objetivo geral desse trabalho é desenvolver um pacote python para facilitar a utilização do Matrix SIM Switch Box (MSSB) na automação de testes móveis, em especial os testes que necessitam da manipulação de cartão SIM. Essa facilidade vai ser obtida através da eliminação da necessidade de escrever código para interagir com a ferramenta, pois o pacote já possui todo o código necessário para isso, bastando apenas ser chamado no script do teste.

Além disso, tem-se os seguintes objetivos específicos:

- Apresentar a ferramenta, descrevendo suas funcionalidades e como utilizar;
- Descrever o pacote python criado;
- Mostrar exemplos de uso do pacote;
- Expor os benefícios que a ferramenta pode prover;
- Identificar limitações para o uso dessa ferramenta.

Existem outros pacotes criados para serem usados em testes automatizados, como o Spliter, usado para testes de aplicações web, o pywinauto, que é usado para automatizar a interface grafica do Windows, permitindo enviar comandos de mouse e teclado para as caixas de diálogo do windows. Outro exemplo de pacote python para teste é o Beautiful Soup, que é usado para extrair dados de arquivos HTML e XML.

Este trabalho é composto por mais 6 seções. A seção 2 descreve a empresa e o ramo em que atua e tambem há quanto tempo eu faço parte da empresa, destacando as funções que atuei e a função atual. A seção 3 apresenta o experimento, explicando as funcionalidades da ferramenta MSSB, as etapas para implementação da ferramenta em um ambiente de automação, a descriçao do pacote python desenvolvido e os testes que foram implementados para usar o pacote criado. Na seção 4 é destacada a minha contribuição e as limitações do trabalho. A seção 5 descreve as dificuldades encontradas durante o desenvolvimento do trabalho. Na seção 6 é apresentado os impactos que a formação na universidade tem no meu trabalho atual. Por fim, a seção 7 diz respeito as conclusões obtidas após a análise da implementação da ferramenta.

2. A Empresa e Sua Atuação

O Centro de Estudos e Sistemas Avançados do Recife, também conhecido por seu acrônimo CESAR, é um centro de inovação com sede na cidade do Recife, Pernambuco, possui regionais em Sorocaba, Curitiba, Manaus, e atendimento comercial em São Paulo, Rio de Janeiro e Flórida(EUA). O CESAR foi fundado em 1996 por três professores do Centro de Informática da UFPE, Silvio Meira, Fábio Silva e Ismar Kaufman, como forma de aproximar a academia do mercado. Tem como missão Identificar, potencializar e concretizar oportunidades de transformação das organizações e da vida das pessoas.

O Cesar trabalha com um time diverso e multidisciplinar de mais de mil colaboradores, incluindo designers, desenvolvedores, consultores, estrategistas, empreendedores, pesquisadores e educadores. Com isso, o Cesar se destaca pela sua capacidade de entrega e excelência nos resultados, gerando bons números como mais de 140 projetos entregues por ano, mais de 20 segmentos atendidos, mais de 80 clientes atendidos por ano e um índice de 79,3% no NPS 2020.1. A Figura 1 mostra alguns dos clientes do Cesar, o que permitiu ao Cesar fechar o ano de 2019 com 133 milhões em vendas.

Figura 1. Alguns dos clientes do Cesar. Fonte: Cesar

O CESAR atua em todo o ciclo de inovação, desde a pesquisa, estudos e formação, evoluindo para a construção de modelos dominantes por meio da experimentação e criação de novos modelos de negócios, chegando até a fase de melhorias incrementais, com o desenvolvimento de soluções robustas e escaláveis. O jeito de inovar do CESAR é inspirado nos métodos clássicos do Design, mas também traz uma abordagem prática fruto da experiência de quem vem trabalhando na solução de problemas complexos há mais de duas décadas. Os eixos de atuação do CESAR são:

- Inovação na educação e competências do futuro;
- Design de produtos e serviços
- Inovação corporativa e estratégia de negócios
- Soluções robustas em TICs

Iniciei minha jornada no Cesar em fevereiro de 2019 como estagiário de testes, sendo efetivado em novembro de 2020 como Engenheiro de Testes I, função que exerço atualmente. Nesse tempo de vínculo pude transitar por alguns projetos, nos quais trabalhei com testes manuais, automação de testes e técnicas de test design como criação de cenários e caso de testes. Atualmente trabalho executando testes manuais em dispositivos wireless com conexão com celulares de sistema Android. Este trabalho foi feito utilizando uma ferramenta do primeiro projeto que trabalhei no Cesar e por questão de confidencialidade, mais detalhes sobre o projeto não serão informados. O mesmo vale para o projeto atual em que atuo.

3. Desenvolvimento na Empresa

O experimento foi feito no eixo de Soluções Robustas em TICs, mais precisamente em um projeto de testes móveis. O projeto, que é de um cliente internacional, possui como principal atividade realizar testes funcionais de algumas features em dispositivos móveis. A Tabela 1 representa o ambiente de execução do experimento.

A equipe do projeto possui pessoas nos seguintes papéis:

- Gerente de projeto;
- Engenheiros de teste;
- Estagiários de engenharia de teste.

Tabela 1. Ambiente do experimento

Empresa	Tipo de Empresa	Objetivo do Projeto
CESAR	Centro de Inovação	Testes funcionais em dispositivos móveis

Fonte: Elaborado pelo autor (2021).

Devido a restrições por confidencialidade, dados sobre os processos usados pelo projeto, bem como os números a respeito dos testes não serão informados nesse experimento.

3.1. A Problemática e Solução Proposta

A automação de testes já é uma realidade do projeto, o qual possui uma boa quantidade dos testes já cobertos pela automação, auxiliando no ganho de tempo para outras demandas que o cliente possa solicitar e também a validar uma maior quantidade de casos de testes em cada ciclo de teste em um aparelho.

Devido a existencia de várias operadoras telefônicas, as quais, algumas delas possuem funcionalidades específicas, é preciso que sejam realizados testes voltados para essas operadoras, e para isso, é necessário que um cartão SIM de determinada operadora esteja inserido no aparelho para realizar o teste, ou ainda pode existir casos em que é necessário remover ou inserir um cartão SIM durante o teste. Para os testes manuais, esse trabalho é feito sem grandes esforços, mas para a automação, essa necessidade de manuseio de cartões SIM tem é um desafio, pois isso limita a quantidade de casos de teste considerados automatizáveis.

Para ajudar com esse problema, foi feita uma pesquisa a respeito de possíveis ferramentas que pudessem simular o manuseio de cartões SIM em um aparelho celular. Como resultado da pesquisa, foi identificado uma única ferramenta que apresentava uma forma de lidar com o problema, o Matrix SIM Switch Box, que foi importado da Alemanha para uso na automação de testes. Para a elaboração deste trabalho, foi realizada uma nova pesquisa mais de um ano após a aquisição do MSSB, para tentar encontrar novas ferramentas que funcionam de forma semelhante ao MSSB, mas não foi encontrada outra ferramenta parecida.

3.2. A Ferramenta Usada na Automação

O equipamento Matrix SIM Switch Box (MSSB) é fabricado pela Qualigon[1] e apresenta tecnologias inteligentes para gerenciar até 32 cartões SIM e até 4 modems ou aparelhos simultaneamente, como ilustrado na Figura 2. O controle completo é feito via USB ou uma API REST usando uma conexão de rede no modo LAN.

Com base em sua arquitetura mestre / escravo com até 8 módulos, cada um contendo até 4 SIMs, esta configuração oferece o mais alto grau de flexibilidade. A Qualigon fabrica o equipamento em 4 modelos:

- MSSB 4x1 multiplexando 4 SIMs para 1 dispositivo;

[1] https://www.qualigon.de/en/test-systems/matrix-sim-switchbox-mssb

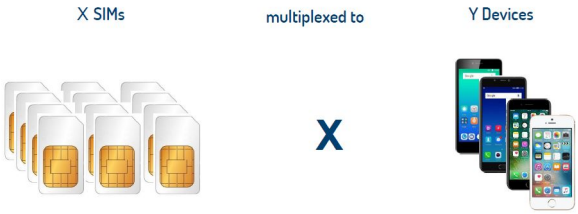


Figura 2. Representação gráfica da ideia principal do MSSB. Fonte: Qualigon

- MSSB 8x4 multiplexando 8 SIMs para 4 dispositivo;
- MSSB 16x2 multiplexando 16 SIMs para 2 dispositivo;
- MSSB 32x1 multiplexando 32 SIMs para 1 dispositivo

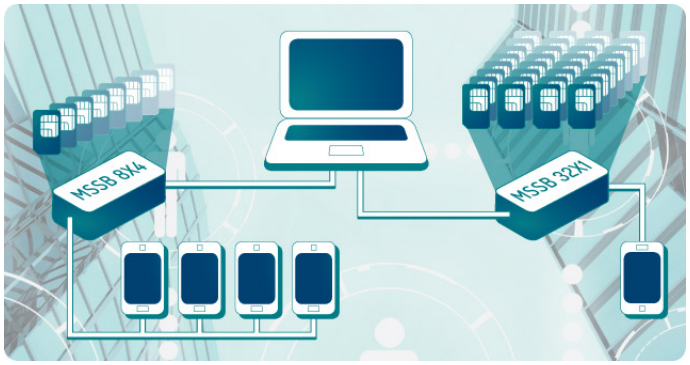


Figura 3. Representação gráfica do ambiente com MSSB. Fonte: Qualigon

A Figura 3 mostra uma ilustração de um ambiente com 2 MSSB devidamente instalados via cabo. O primeiro com suporte para 8 cartões SIM, ligado a 4 celulares e o segundo com suporte para 32 cartões SIM, ligado a apenas um celular. Ambos os equipamentos são conectados a um computador, o qual exibe a interface para controle do MSSB.

Todos os MSSB podem ser preparados para configuração LOCAL ou LAN. Para acesso local ou direto, você pode controlar o MSSB por meio da interface USB. Em configurações remotas, você pode controlar o dispositivo MSSB também via LAN e usar a interface gráfica do usuário ou comandos HTTP para selecionar os SIMs. O MSSB 4x1 pode ser controlado via bluetooth para facilitar o uso.

O MSSB fornece um número de funcionalidades que possibilitam obter benefícios a partir delas, dentre essas funcionalidades, estão:

- Multiplexador combinado de “múltiplos SIM, múltiplos terminais”;
- Troca controlada por software de até 32 cartões SIM e até 4 dispositivos USB;
- Utilizável em cenários LOCAL e LAN com acesso USB ou LAN;
- Fácil acesso aos cartões SIM;
- Fácil integração em sistemas existentes;
- Sistemas operacionais com suporte Windows® 10, 8.x, 7, XP, Linux, Mac OS X®, Android;
- Interfaces de controle de software sofisticadas via DLL, JAVA, C / C ++ e HTTP.

Matrix SIM Switch box Admin é o painel baseado na web para controlar o MSSB. Neste painel, é possível atribuir os SIMs em uma interface gráfica de usuário (GUI) aos dispositivos móveis instalados. O MSSB Admin também oferece suporte para carregar os SIMs e definir nomes fáceis de usar para o uso diário:

- MSSB Admin oferece controle baseado em host do dispositivo MSSB;
- Funcionalidade para reiniciar os dispositivos;
- A interface do software fornece controle externo de todas as funcionalidades e recursos principais do MSSB;
- Integração amigável e econômica em sistemas existentes.

Figura 4. Interface web da versão 8x4 para remoção e inserção de sim card no aparelho celular. Fonte: Qualigon

A Figura 4 ilustra a interface na qual pode ser feita a inserção ou remoção de um cartão SIM no celular. Essa interface corresponde a versão de 8x4 da ferramenta.

Matrix SIM Switch box se adapta perfeitamente aos processos de automação de testes. É possível controlar o MSSB com comandos diretamente de seu sistema de desenvolvimento. Entre as funcionalidades disponíveis para automação estão:

- Troca de SIM real, sem virtualização de SIM;
- Acesso total aos SIMs suportados;
- Troca de celulares baseada em host em cenários de teste;
- Garantia de comunicação máquina a máquina (M2M) via seleção de rede.

Para a utilização do MSSB, o primeiro passo foi realizar testes manuais com a ferramenta, verificando todo o hardware, que é composto pela placa que recebe os cartões SIM, uma raspberry que roda o serviço web, os adaptadores que são inseridos na bandeja de cartão SIM do aparelho, este ilustrado na Figura 5, e os cabos necessários. A Figura 8 mostra o MSSB ligado conectado a um celular.

Figura 5. Adaptador que é inserido no celular e se comunica com a placa do MSSB. Fonte: Qualigon

Os adaptadores são peças importantes, pois são eles os responsaveis por substituir um cartão SIM no aparelho. São formados por um conector, um cabo flex fino o suficiente para que seja possível fechar a bandeja de cartão SIM do aparelho e um cartão para substituir um cartão SIM real. As Figuras 6 e 7 ilustram como o adaptador é conectado ao aparelho.

Figura 6. Adaptador inserido no aparelho. Fonte: Qualigon

Figura 7. Adaptador posicionado na bandeja de cartão SIM do aparelho. Fonte: Qualigon

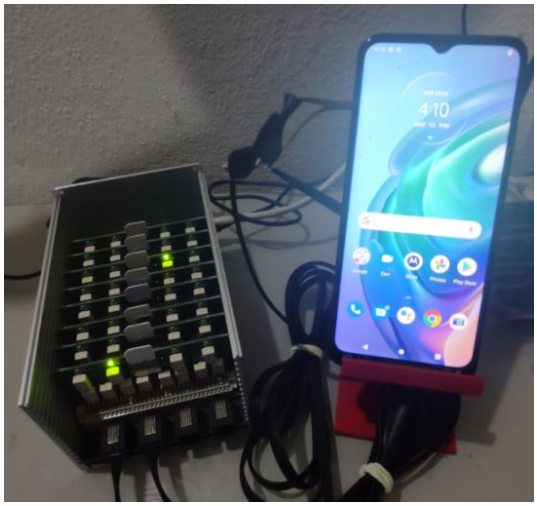


Figura 8. MSSB conectado ao aparelho celular com 2 cartões SIM selecionados. Fonte: Elaborada pelo autor

Com a instalação feita, como mostrado na Figura 8 incluindo a inserção dos cartões SIM na placa, foi possível interagir com a ferramenta através de sua interface web. Nessa interface, é possível cadastrar os dados a respeito dos cartões SIM inseridos, como mostrado na Figura 9, bem como cadastrar os dados dos slots dos celulares, sendo eles single ou dual sim, mostrado na Figura 10.

Figura 9. Interface web para cadastro dos cartões SIM inseridos no MSSB. Fonte: Qualigon

Em seguida, ainda na interface, foi possível manipular os cartões SIM inseridos no MSSB para inserir e remover no aparelho. A inserção e remoção funcionam no modo arrastar e soltar, sendo possivel selecionar um cartão SIM disponivel no MSSB e arrasta-lo através da interface até o aparelho que se deseja inserir o cartão e para remover, basta fazer o procedimento reverso. Esse procedimento é ilustrado na Figura 11

Após essa validação manual, foram criados scripts de requests para a API do MSSB, onde esses requests executam a ação de inserir ou remover um cartão SIM do

Figura 10. Interface web para cadastro dos aparelhos. Fonte: Qualigon

Figura 11. Interface web para inserção e remoção dos cartões SIM do aparelho. Fonte: Qualigon

celular. A identificação do cartão SIM pode ser feita através de valores únicos de cada cartão, como o IMSI.

IMSI significa International Mobile Subscriber Identity. É um número exclusivo que os Operadores de Rede Móvel (MNOs) usam para reconhecer assinantes individuais e é um componente-chave de um perfil do Módulo de Identidade do Assinante (SIM)[EMnify 2020].

O IMSI possui o seguinte formato:

$$IMSI = MCC + MNC + MSIN, \tag{1}$$

Onde:

- MCC = Mobile Country Code (código do país do celular);
- MNC = Mobile Network Code (código da rede celular);

- MSIN = Mobile Station Identification Number (número de identificação do celular).

  Exemplo:

$$IMSI = 311(MCC) + 030(MNC) + 000002359(MSIN) = \mathbf{311030000002359} \quad (2)$$

3.3. Solução Proposta Para Interação com o MSSB

Para facilitar o uso do MSSB na automação, foi criado um pacote python, chamado mssbox, com funções necessárias para interação com a ferramenta, que são as funções de conectar e desconectar um cartão SIM. O pacote pode ser instalado em seu ambiente de automação, possibilitando o uso de seus métodos para manipular os cartões SIM inseridos na ferramenta. A Figura 12 mostra como o pacote foi estruturado. Os arquivos LICENCE.txt, README.rst, setup.cfg e setup.py são os arquivos necessários para a criação de um pacote python. O setup.py contém todas as informações do pacote, como versão. descrição, autor, outros pacotes necessários(caso possua algu requisito), entre outras informações. Dentro do diretório mssb estão os arquivos python com as funções que interagem com o MSSB, sendo os principais o connect.py e o disconnect.py, além de outros arquivos dentro de ”util”com algumas funções auxiliares.

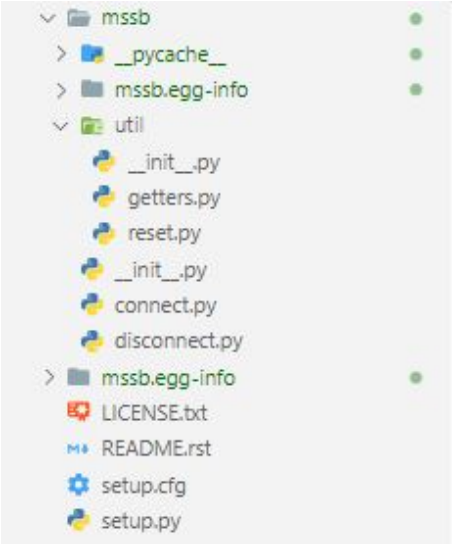


Figura 12. Estrutura do pacote mssb. Fonte: Elaborada pelo autor

Ao adicionar os cartões no MSSB, através da interface web no navegador é possível inserir os dados do cartão, e após isso, usar o botão ”exportar”na interface para gerar um arquivo com essas informações. Para utilização do pacote, é necessário que este arquivo seja colocado em um diretório de fácil acesso, podendo ser uma pasta específica dentro do ambiente de automação. Além disso, é necessário conhecer o endereço IP e o serial do MSSB, pois eles devem ser utilizados ao chamar os métodos do pacote.

O pacote é formado por dois arquivos principais, o connect.py e o disconnect.py, que são responsáveis por inserir e remover, respectivamente, um cartão SIM da ferramenta. Além dos principais, outros arquivos auxiliares foram criados, com métodos que auxiliam na inserção ou resetam a ferramenta. Um arquivo auxiliar é o reset.py, com

método para resetar o MSSB, desconectando todos os cartões SIM da ferramenta. A Figura 13 mostra o código criado para resetar o MSSB.

```python
import requests
import xml.etree.ElementTree as ET


def reset_all_connections(mssb_ip, mssb_serial):  # PAY ATTENTION! THIS METHOD IMPACTS ALL DEVICES CONNECTED TO THE MSSB
    """
    This method disconnects all SIMs connected to any terminals
    :return: returns the result of the request, 'true' or 'false
    :rtype: bool
    """
    response = requests.get(f"http://{mssb_ip}/rest.php?command=resetall&mssbSerial={mssb_serial}")
    response_xml = ET.fromstring(response.content)
    success_value = response_xml.find('params').find("success").text
    return bool(success_value)

```

Figura 13. Código responsável por resetar o MSSB. Fonte: Elaborada pelo autor

O arquivo getters.py é composto por três métodos do tipo ”get”, responsáveis por obter dados do MSSB ou dos cartões SIM que estão inseridos na ferramenta. A Figura 14 ilustra duas funções presentes no arquivo getters.py, a primeira função retorna o cartão SIM que está conectado ao terminal informado, e para isso, utiliza a segunda função que retorna todos os cartões SIM que estão inseridos no MSSB.

```python
def get_connected_sim(mssb_ip, mssb_serial, slot_terminal):
    """
    This method returns the SIM Card allocated to the provided slot
    :param slot_terminal: int MSSB terminal port assigned to the slot
    :return: returns a dict with SIM attribs (id, name, custom_1 (IMSI), custom_2 (PLMN)
    """
    sim = get_all_connected_sims(mssb_ip, mssb_serial).get(int(slot_terminal))
    return sim


def get_all_connected_sims(mssb_ip, mssb_serial):
    """
    This method returns all SIMs connected to terminals
    :return: A dict with all SIMs with their attribs (id, name, custom_1 (IMSI), custom_2 (PLMN)
    """
    response = requests.get(f"http://{mssb_ip}/rest.php?command=listcon&mssbSerial={mssb_serial}")
    response_xml = ET.fromstring(response.content)
    connections = response_xml.find('connections').findall("terminal")
    sims = dict()
    for i in range(0, len(connections)):
        sims[i] = connections[i].find("sim")
    return sims
```

Figura 14. Funções utilizadas para obter qual cartão SIM está connectado a determinado terminal da ferramenta. Fonte: Elaborada pelo autor

Uma outra função auxiliar implementada foi o ”get_sims_info”. Essa função vai retornar um dicionário com todos os dados dos cartões SIM inseridos na ferramenta. Ela pode ser usada quando ainda não se tem registrado as informações dos cartões SIM inseridos na ferramenta. Para utilização dessa função, é necessário informar o caminho para o arquivo exportado através da interface do MSSB, pois essa função irá ler o arquivo e montar o dicionário de retorno. A função pode ser vista a seguir na Figura 15

No arquivo connect.py, foi implementado o código responsável por conectar um cartão SIM escolhido a determinado terminal, como pode ser visto na Figura 16. O código utiliza um dicionário com todos os dados dos cartões inseridos na ferramenta, quando esse dicionário não é informado, a função get_sims_info é chamada para que o dicionário seja criado a partir do arquivo exportado na interface web do MSSB.

Nesta função, é possível escolher um cartão SIM aleatório dentre os que estão inseridos na ferramenta e também é possível restringir a escolha do cartão por operadora ou escolhendo um cartão específico. Para restringir por operadora, basta passar como

```python
def get_sims_info(repos_dir, mssb_serial, mssb_file_dir):
    """
    This function gets the info for the SIMs connected to the MSSB via file exported from MSSB
    :param repos_dir: Your repository directory
    :param mssb_serial: Serial for the MSSB
    :param mssb_file_dir: The directory with the mssb file
    :return: dict with the SIMs Config
    """
    file = f"{repos_dir}/{mssb_serial}_sims.json"
    mssb_file = f"{mssb_file_dir}/{mssb_serial}"
    with open(file, 'wb') as f:
        f.write(mssb_file.content)
    with open(file) as json_file:
        data = json.load(json_file)
    return data
```

Figura 15. Função para obtenção dos dados de um cartão SIM inserido no MSSB. Fonte: Elaborada pelo autor

```python
def connect_sim(mssb_ip, mssb_serial,
                repos_dir, mssb_file_dir,
                sim_plmn, slot_terminal,
                sims_config=None, sim_imsi=None):
    """
    This method connects the desired SIM having that PLMN to the provided slot/terminal
    :param repos_dir: Your repository directory
    :param mssb_file_dir: The directory with the mssb file
    :param sim_plmn: PLMN of the desired SIM Card
    :param slot_terminal: slot_terminal: MSSB terminal ID port assigned to the slot
    :param sims_config: dict containing the sims_config
    :param sim_imsi: IMSI for the specific SIM Card to be connected
    :return: returns whether the operation succeeded or not
    :rtype: bool
    """
    if sims_config:
        sims = sims_config
    else:
        sims = get_sims_info(repos_dir, mssb_serial, mssb_file_dir)
    sim_mssb = 0
    for i in range(0, len(sims)):
        if sim_plmn == sims[i].get('DEV_CUSTOM_2'):
            if sim_imsi:
                if sim_imsi == sims[i].get('DEV_CUSTOM_1'):
                    sim_mssb = sims[i].get('ID')
            else:
                sim_mssb = sims[i].get('ID')
            break
    response = requests.get('http://' + mssb_ip + '/rest.php?'
                                                  'command=connect'
                                                  '&sim='+str(sim_mssb)+
                                                  '&terminal='+str(slot_terminal)+
                                                  '&mssbSerial=' + mssb_serial)
    response_xml = ET.fromstring(response.content)
    sim_success_value = response_xml.find('params').find("success").text
    connected = get_connected_sim(mssb_ip, mssb_serial, int(slot_terminal)).get('custom_2') == sim_plmn
    return sim_success_value == "1" and connected
```

Figura 16. Função para inserir um cartão SIM no celular. Fonte: Elaborada pelo autor

parametro para a função o PLMN, que é um valor único de cada operadora presente no cartão SIM. Já para escolher um cartão SIM específico, basta informar o IMSI, dado que esse valor é único de cada cartão. O IMSI também é aplicável quando se deseja inserir 2 cartões da mesma operadora, pois é através do IMSI que a função vai diferenciar os cartões quando o mesmo plmn for informado.

Para remover um cartão SIM de um aparelho, a função disconnect_sim pode ser chamada, recebendo como parametro os dados do MSSB e qual o terminal que vai ter o cartão retirado. Essa função pode ser vista na Figura 17.

```python
def disconnect_sim(mssb_ip, mssb_serial, slot_terminal):
    """
    This method disconnects any SIM attached to the specified terminal/slot
    :param slot_terminal: MSSB terminal ID port assigned to the slot
    :return: returns whether the operation succeeded or not
    :rtype: bool
    """
    if get_connected_sim(mssb_ip, mssb_serial, int(slot_terminal)).get('id') == "":
        return True
    response = requests.get(f"http://{mssb_ip}/rest.php?command=disconnect&sim=0&terminal={str(slot_terminal)}&mssbSerial={mssb_serial}")
    response_xml = ET.fromstring(response.content)
    sim_success_value = response_xml.find('params').find("success").text
    return sim_success_value == "1" and get_connected_sim(int(slot_terminal)).get('id') == ""
```

Figura 17. Função para remover um cartão SIM do celular. Fonte: Elaborada pelo autor

3.4. Testes Realizados

Para exemplificar o uso do pacote criado, foi criado um pequeno projeto em python para testes em celulares Android da Motorola, utilizando o Appium, que é um framework de automação de testes open source para uso com web apps nativos, hibridos e móveis, como descrito em seu site[2].

Appium é uma estrutura de automação de teste de código aberto para uso com aplicativos da Web nativos, híbridos e móveis. Ele pode ser utilizado para aplicativos iOS, Android e Windows usando o protocolo WebDriver.

O Appium baseia-se na ideia de que testar aplicativos nativos não deve exigir a inclusão de um SDK ou a recompilação de seu aplicativo. E que você deve ser capaz de usar suas práticas de teste, estruturas e ferramentas preferidas. Appium é um projeto de código aberto e tomou decisões de design e ferramentas para incentivar uma comunidade vibrante de contribuições.

Para fins demonstrativos, esse projeto contem 2 testes automatizados e as informações dos cartões SIM inseridos na ferramenta foram obtidos através da interface web e foram colocadas em um dicionario no arquivo ”config.py”. A Figura 18 mostra a estrutura do projeto.

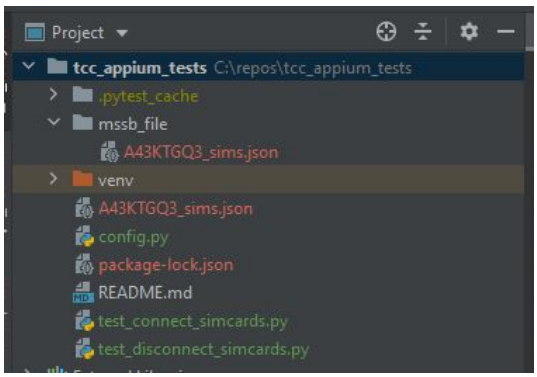


Figura 18. Estrutura do projeto de testes automatizados para uso do pacote mssbox. Fonte: Elaborada pelo autor

Além do arquivo ”config.py”, o projeto também conta com o arquivo json exportado pela interface web, de nome ”A43KTGQ3_sims”e localizado dentro do diretório ”mssb_file”. É através desse arquivo que o pacote vai conseguir coletar os dados de todos os cartões SIM que estão inseridos no MSSB, caso esses dados ainda não sejam conhecidos, diferente do que foi implementado, em que os dados já são conhecidos.

Exemplificando o uso do pacote em testes automatizados, a Figura 19 mostra um trecho do código de um teste que valida na tela de configuração de cartão SIM que um cartão está ativo. O teste utiliza o pacote criado para inserir 2 cartões SIM no aparelho e faz a validação que o cartão inserido está ativo no aparelho. Na figura é mostrada a primeira parte do teste, que insere o primeiro cartão no slot 1 do aparelho e valida na tela de configuração que o cartão do slot 1 está ativo e o do slot 2 está inativo. O trecho de código partindo da linha 15 até a 28 se repete em seguida para inserir o segundo cartão

[2]https://appium.io

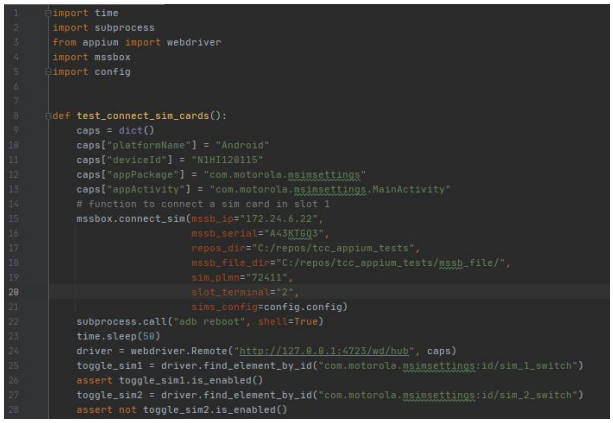


Figura 19. Trecho do código do teste que insere cartões SIM no aparelho e valida que estão ativados. Fonte: Elaborada pelo autor

SIM no slot 2 do aparelho e validar na tela de configuração que os 2 slots estão ativos. A Figura 20 mostra a tela do celular em que a validação é feita e essa validação é executada em 2 momentos sendo a tela da esquerda validada ao inserir o primeiro cartão SIM e a tela da direita quando o segundo cartão SIM é inserido no aparelho.

Um outro teste valida a remoção de todos os cartões SIM do aparelho e verifica na tela já mostrada que todos os slots estão desativados. Como já citado anteriormente, não é necessário dar um reset no aprelho para que a remoção seja feita, portanto o código usa o pacote para remover os cartões e logo em seguida já abre a tela de configuração de cartão SIM para validar que foram removidos. A Figura 21 mostra o código desse segundo teste.

4. Contribuição e Limitações do Trabalho

Esse pacote foi disponibilizado no PyPI [Silva 2022], que é um repositório de software para a linguagem de programação Python, no qual pode ser encontrado diversos outros pacotes. Com a publicação no Pypi, o pacote criado neste trabalho pode ser instalado através do comando 'pip install', possibilitando que a comunidade também tenha acesso ao pacote, podendo ser utilizado em outros cenários e empresas, caso também utilizem o MSSB.

4.1. Limitação dos celulares

Uma limitação observada durante os testes foi a de que é necessário reiniciar o modem do celular após inserir um cartão SIM. Isso pode ser feito reiniciando o aparelho, logo a necessidade de se mandar um comando de reboot para o aparelho logo após inserir um cartão acaba impedindo a automação dos testes que necessitam de um *hot swap*, que é a troca de cartão SIM com o aparelho ligado. Essa limitação não foi observada para a remoção do cartão, onde apenas o request de remover é necessário para o que o celular não reconheça mais o cartão que estava inserido.

5. Dificuldades Encontradas

A primeira dificuldade encontrada foi a restrição de dados devido a acordo de confidencialidade com o cliente, logo não pude aproveitar códigos desenvolvidos no projeto, sendo necessário que eu mesmo criasse os testes para uso do pacote. Outra dificuldade

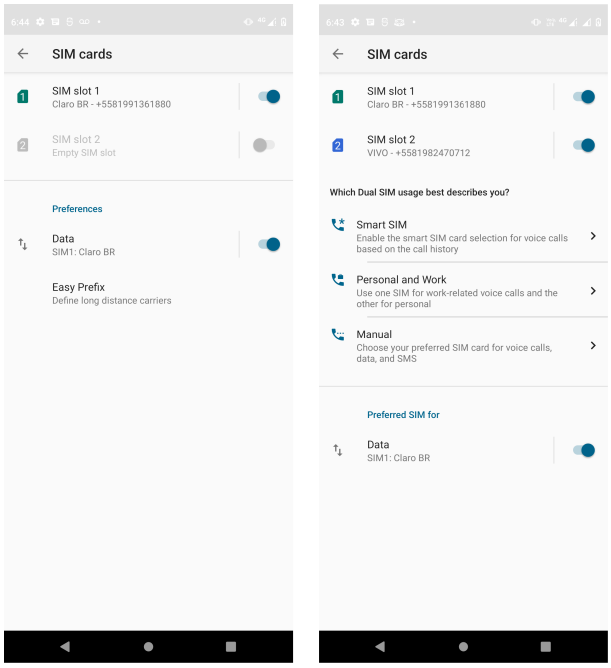


Figura 20. Tela do Android em que é feita a validação que o cartão SIM foi inserido e está ativo. Fonte: Elaborada pelo autor

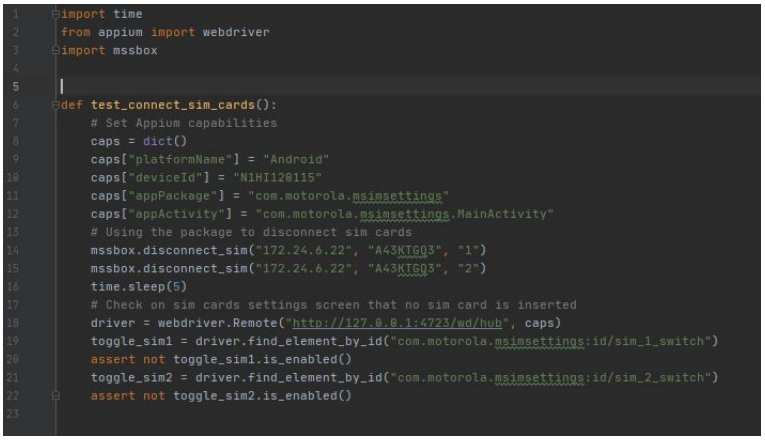
foi a de acesso a ferramenta MSSB, devido a minha mudança de projeto dentro da empresa, deixando de trabalhar diretamente com o MSSB. Para contornar isso, tive a ajuda do co-orientador Renan para ter acesso a ferramenta e realizar os testes necessários. Para o desenvolvimento do pacote, a maior dificuldade foi com a estrutura do pacote, de como ele deve ser organizado, quais os arquivos necessários e que tipo de dados devem está presente em cada arquivo para que seja possível publicar ele no Pypi e que a partir dessa pulicação, o pacote seja utilizável em outros projetos.

6. Impactos da Formação no Trabalho

A formação tem sido importante para minha atuação na empresa, pois tenho conseguido por em prática os conhecimentos de programação adquiridos nas disciplinas que envolviam desenvolver projetos e aplicar os conceitos de programação na prática, nas quais o foco é na linguagem python e nesses projetos em que atuei no Cesar, o python está presente na automação de testes. Outro ponto importante foi a disciplina de Testes de Software, onde pude ter contato com a teoria e fundamentos do teste de software, os vários tipos de teste e em que situação são necessários, além de entender a importancia do teste dentro do ciclo de desenvolvimento de um sistema. Todo esse conhecimento adquirido nessa disciplina me ajudou a entender melhor a área em que atuo hoje e pretendo seguir carreira.

7. Conclusões

Como foi visto, os aparelhos celulares tem feito cada vez mais parte da vida em nossa sociedade, devido a alta tecnologia empregada nestes dispositivos e no quanto essas tecnologias auxiliam para tarefas comuns do dia a dia. Uma prova disso está no alto número de aparelhos celulares vendidos nos ultimos anos, bem como na alta variedade de aparelhos disponíveis no mercado.



```python
import time
from appium import webdriver
import mssbox

def test_connect_sim_cards():
    # Set Appium capabilities
    caps = dict()
    caps["platformName"] = "Android"
    caps["deviceId"] = "N1HI120115"
    caps["appPackage"] = "com.motorola.msimsettings"
    caps["appActivity"] = "com.motorola.msimsettings.MainActivity"
    # Using the package to disconnect sim cards
    mssbox.disconnect_sim("172.24.6.22", "A43KTGQ3", "1")
    mssbox.disconnect_sim("172.24.6.22", "A43KTGQ3", "2")
    time.sleep(5)
    # Check on sim cards settings screen that no sim card is inserted
    driver = webdriver.Remote("http://127.0.0.1:4723/wd/hub", caps)
    toggle_sim1 = driver.find_element_by_id("com.motorola.msimsettings:id/sim_1_switch")
    assert not toggle_sim1.is_enabled()
    toggle_sim2 = driver.find_element_by_id("com.motorola.msimsettings:id/sim_2_switch")
    assert not toggle_sim2.is_enabled()
```

Figura 21. Código do teste que valida que todos os cartões SIM foram removidos do aparelho.. Fonte: Elaborada pelo autor

Por ser algo já tão difundido entre os usuários, é necessário garantir que esses produtos tenham qualidade, e uma forma de fazer isso, é através dos testes. Restrigindo um pouco para apenas o software desses dispositivos, como saber se esse software está sendo testado corretamente? Ou ainda, quais os desafios para se realizar testes em dispositivos móveis?

Um ponto levado em consideração neste trabalho foi o de que o software embarcado no aparelho pode possuir algumas funcionalidades que precisam de um cartão SIM para serem habilitadas, dessa forma, é necessária interação humana para manipulação dos cartões SIM e a realização correta dos testes. Essa necessidade de manipulação de cartões de certa forma é um desafio para se automatizar os testes destas funcionalidades especificas.

Para o entendimento deste trabalho, foi realizada uma pesquisa bibliográfica a respeito dos principais conceitos necessários, partindo da engenharia de software como mais geral, até ir aprofudando nos conceitos de qualidade de software, testes de software e automação de teste.

Devido a restrições por acordo de confidencialidade, não foi possível expor dados referentes ao projeto alvo do experimento, como dados a respeito do processo de teste utilizado, bem como os dados referente a automação de testes utilizada. Dessa forma, o foco maior do trabalho além de apresentar a ferramenta Matrix SIM Switch Box e os seus benefícios, foi utilizar a ferramenta para automatizar alguns testes genéricos para aparelhos com sistema Android.

Como apresentado no trabalho, a ferramenta Matrix SIM Switch Box ajuda na problemática proposta, pois ela funciona fazendo o gerenciamento de cartões SIM no aparelho celular, eliminando a necessidade de interação humana para a troca de um cartão SIM do aparelho, possibilitando que testes que realizam a inserção ou remoção de um cartão SIM possam ser automatizados. O equipamento traz uma solução robusta, onde o controle completo pode ser feito via USB ou por meio de uma API REST, usando uma rede LAN.

A depender do modelo escolhido do MSSB, é possível ter o controle de até 32

cartões SIM e até 4 dispositivos USB, onde cada dispositivo USB corresponde a um slot de cartão SIM de um aparelho celular. Além disso, entre as funcionalidades estão presentes o fácil acesso aos cartões, a fácil integração aos sistemas existentes e também a sua interface, que fornece controle externo de todas as funcionalidades e recursos principais do MSSB.

Para automação, foi criado um pacote python chamado "mssbox", o qual possui métodos que realizam requests para a API da ferramenta. Através desses métodos, é possível inserir ou remover um cartão SIM do aparelho celular durante a execução de um teste automatizado, ou seja, é possível que as chamadas para estes métodos sejam incluídas em um script de teste já definido, permitindo que os testes que realizam a troca de um cartão SIM no celular sejam também automatizados. O pacote criado ajuda na montagem do ambiente de teste, pois dispensa a implementação de código para interação com o MSSB, permitindo que as pessoas envolvidas em seu uso se preocupem apenas com o código dos testes e da arquitetura utilizada.

Para a utilização do pacote, é necessário conhecer o endereço IP e também o serial do MSSB, pois esses dados devem ser passados como parametros para os métodos do pacote. Também é necessário ter as dados dos cartões inseridos na ferramenta, que podem ser obtidos exportando um arquivo através da interface web. Este arquivo deve ficar em um diretório de fácil acesso, pois caminho desse arquivo também deve ser utlizado para realizar as chamadas do pacote.

O pacote foi disponibilizado no pypi, o repositório da linguagem python, dessa forma, qualquer pessoa que possua uma unidade do MSSB e deseja automatizar testes móveis pode fazer a instalação do pacote em seu ambiente e utilizar os seus métodos. Para utilizar o pacote criado, foram desenvolvidos 2 testes automatizados em python utilizando o framework Appium, nesses testes, pode-se perceber o funcionamento do pacote, conectando e desconectando um cartão SIM do aparelho e fazendo as validações necessárias após essa operação.

Com a implementação do MSSB, é possível ter um aumento na quantidade de testes automatizáveis, pois a necessidade de interação humana para manipulação dos cartões SIM é em grande parte resolvida. Como consequência do aumento dos testes automatizáveis, também pode-se obter um aumento na quantidade de testes automatizados, bem como um aumento na quantidade de testes executados, pois com o MSSB, é possível definir um setup até para os testes que não realizam inserção ou remoção de cartão SIM do celular, ou seja, ainda no setup, ele insere o cartão necessário no aparelho e em seguida todos os testes para aquele cartão são executados, eliminando o tempo de interação humana para fazer esse setup manualmente.

Referências

Amalfitano, D., Fasolino, A. R., Tramontana, P., and Robbins, B. (2013). *Testing Android Mobile Applications: Challenges, Strategies, and Approaches*, volume 89, pages 1–52. Academic Press Inc.

Carvalho, F. (2019). Tecnologia mobile é a cultura do novo século. https://itforum.com.br/colunas/tecnologia-mobile-e-a-cultura-do-novo-seculo/. Data de acesso: 10 dez. 2021.

EMnify (2020). What is an international mobile subscriber identity (imsi)? https://www.emnify.com/iot-glossary/imsi. Data de acesso: 8 dez. 2021.

IBM (2021). What is mobile technology? https://www.ibm.com/topics/mobile-technology. Data de acesso: 10 dez. 2021.

In, B. (2021). Mobile technology. https://builtin.com/mobile-technology. Data de acesso: 13 dez. 2021.

O'Dea, S. (2021). Number of smartphones sold to end users worldwide from 2007 to 2021. https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/statisticContainer. Data de acesso: 8 dez. 2021.

Silva, A. (2022). mssbox 1.0.7. https://pypi.org/project/mssbox/. Data de acesso: 26 mai. 2022.