PUMA automation framework design

* Ralf Wang

# Introduction

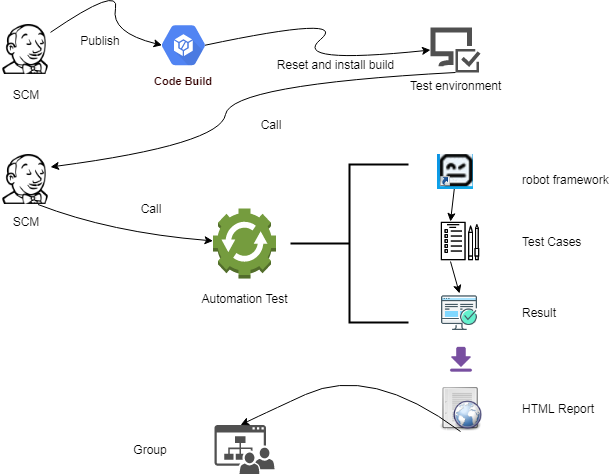
The China Kiosk R&D quality team decide to do the automation testing. We plan to use python and robot framework to realize the requirement and design.

Team plan to integrate with Jenkins to make the workflow automatically. Then we can verify the build and patch quickly and automatically. In first phase, we will realize the build deploy and BVT test function before the end of this year.

**Architecture**

The automation main workflow is that:

1. Jenkins deploy the build/patch in system.
2. Jenkins jobs install the build/patch in test environment with application or tool.
3. If the installation is successfully, it will start a jobs to execute the automation test scripts.
4. The jobs will collect the result and send the result to stakeholder.



For first phase, there are two features we need to publish: CI/CD and BVT test.

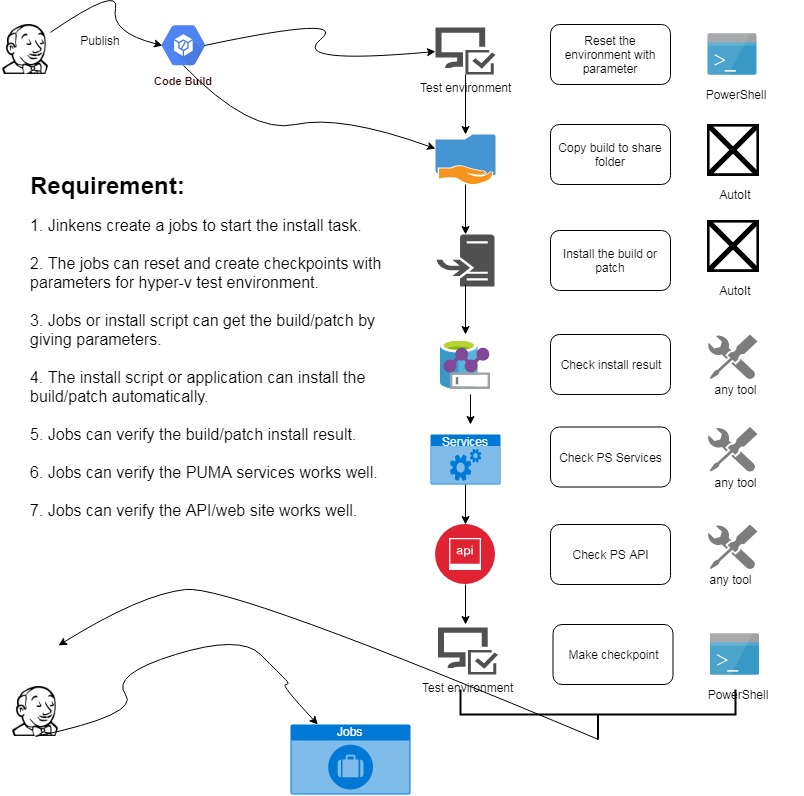
**CI/CD:**

SCM team will in charge of the build/patch deploy and install works.

Before start the detail works, team need do some prepare works as follow:

1. Prepare a test environment which include Hyper-V software. --QA
2. Prepare a script or application which can download and install the build/patch automatically. –SCM
3. Ensure the Jenkins jobs can reset/create OS checkpoint with parameters. –SCM
4. Prepare scripts to verify the installation result. –SCM
5. Prepare scripts to verify the services running status. –SCM
6. Prepare scripts to verify the API and web site works well. –SCM

At this feature, the workflow as follow:



**Analyze:**

**Test environment:**

We used to use the PowerShell to reset and create checkpoint in history automation script with C#. This function can also realize with python scripts. The PowerShell can also execute command by remote mode. Detail information can refer the document as <https://www.cnblogs.com/wanghao4023030/p/11151099.html>

**File download:**

There should have a script or application can download the deployed build/patch. The file name and version should have constant name rule. The file name should transfer to script as variable then it can install and record the log.

**Installation:**

The application or script should can install the build/patch automatically. Ensure the application can identify the installation package is build or patch. That means the application can install build and patch. Team has the experiments of it which design by the Auto It tool. SCM team can ask help from Gavin Chen.

**Installation Check:**

After the installation works, there should have a script to do correctness verification operation. Ensure the build or patch is install successfully.

**Service Check:**

After the installation works, there should have a script to check the related services work well such as PS services, IIS, SQL server services etc. The detail information as follow:

1. PS services: Check all services start successfully. The service list can get from the ‘LCPconfig.dat’.
2. IIS: Ensure the IIS service works well
3. SQL servers: Ensure the SQL Server (GCPACSWS), SQL server Browser works well. The remote client can connect the database by TCP/IP protocol.

**API Check:**

After the installation works, there should have a script to check the related web services work well such as integration services, platform etc. The detail information as follow:

1. Integration services. Ensure all integration web service can connect successfully.
2. API: Ensure all API services works well. Can get the detail list in IIS control panel.

**Log:**

Please ensure that all operations should record with log file. If there happens some issues, the Jenkins jobs can collect the log files to help team identify the root cause. Write in one log files is better choice for the framework.

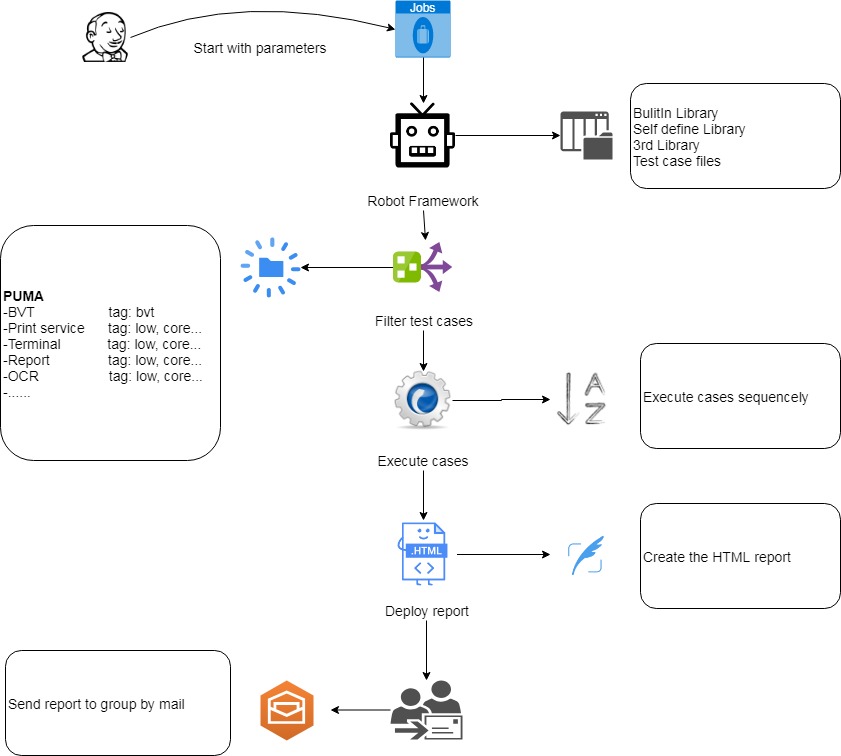
**BVT:**

In the first phase, we will automated the BVT related test cases. These cases include the basic work flow of patient create, report archived and print, film archived and print and other feature which team defined. The test cases will design by the python scripts and organize with the robot framework. Jenkins jobs will trigger the execute operations with parameters.

Before we design the test cases, we should define and do some prepare works.

1. Define the Python version and install in the environment. (Python 3.7.4) --Ralf
2. Define the robot version and create the test environment. 1.7.3.1 --Ralf
3. Make the scripts management rule. --Ralf
4. Define the test cases management rule and strut. --Ralf
5. Define the cases tag list for framework to filter cases. --Ralf
6. Confirm the test cases of BVT. --Team

The expected cases execute process as follow:



**Jobs:**

The jobs should start with parameters to call the python and robot framework. Then framework can execute the selection cases and publish the result in defined location.

**Robot:**

Team should import the required library, 3rd library and self-define library in the frameworks. The test cases related files, resource also should organized in the same folder.

**Test cases:**

The test cases should organized with the suite. Means we will create the multiple suites to cover each features of QC. The suites structure should same as QC. Each case should have one or multiple tags. The robot can execute these cases with conditions. The case name should clearly, readable such as Module\_Name\_TestcaseID. The manual test cases steps should include in the document content in the case file.

**Case execute:**

The test cases should execute with single process because our product`s peculiarity. The case should have clearly log information’s. It will be helpful for us to identify the issues of scripts. Add more exception catch and self-exception pop code in the scripts.

**HTML Report:**

We plan use the robot framework default test log for our report file. If team have enhancement of this. We will re-design it to have more content or readable.

**Mail:**

If the automation script execute finished or have issues, the Jenkins jobs should send the result to stakeholders with mail. The mail members should can configuration in Jenkins.

**BVT Module:**

Follow exist BVT test cases, we investigate the PUMA system and filter these modules we should to design: SCM, Server, Patient, Film, Report, Print mode, holding time, Terminal.

We should design some related functions or methods to realize our test cases. Detail contents please see the document as follow:



The document should upload to SVN server.

We will use the http API to realize some functions. There are some API documents as reference as follow:

**Code Standards**

The team make a code standards for design the automation script or application as follow:

1. The Python code should meets the requirement of PEP8. The detail information can refer the document with URL: <https://github.com/python/peps/blob/master/pep-0008.txt>
2. The feature of PS system should map to a python class. The detail operation should design as the method of the object.
3. All known parameter and variable value should design as the local file. These data type should design as Json.
4. All methods of object`s return data should be Json type.
5. All assisted module, file, tool of one class should organize with one folder.
6. The self-define library should map to the feature in QC and name is begin with ‘PUMA’. Such as PUMA\_Patient\_Library.
7. The class method will be key words in library, the function name should begin with the class name. Such as ‘Patient\_create\_randomly’
8. There are should have document or function help content in the self-define library key words.
9. Every self-define function should add comments and sign developer name

Add by 2019-11-28

**Tool version：**

Python: 3.7.4

Robot Framework ride: RIDE 1.7.3.1

**Case management:**

Now, the smoke automation test case have updated in the PLI system. And path is PUMA -> SmokeTest -> autoworkflow.

Other automation test cases will be review and update after team decide to do next phase work.

**Code management:**

The code will archived in the SVN server: https://shgwp1065/svn/kiosk/documents/QA Team/PUMA\_AUTO. The rule which how to merage the codes with multiple users will realease later.

**Project management:**

We will use the KANBAN system to manage the project. Please register a account in trello and join the project: <https://trello.com/b/zct32H1F/puma-automation> . Notice: Please use the self mailbox to register the account.

**Test system:**

Server: 10.112.20.84

User/password: administrator/pdchi2002$

System: PS 3.0.4.3100

**Test Framework prepared work:**

Python code and robot framework need some 3rd libray during exeuting. The list as follow:

The red need install by user, suggest to install in the global environmnet, Black one include in the python builtin system, do not need install. PIP install command like: pip install package name

* Datetime
* requests
* beautifulsoup4
* pyodbc
* xmltodict
* suds-jurko
* uuid
* configparser
* decimal
* Json
* os
* subprocess
* glob
* time
* collections
* random
* re
* operator
* win32api
* win32print
* sys
* shutil

**Other settings**

1. Need add two terminalfor test, the settings refer PS 10.112.20.84
2. OCR need define some rule to mneet the automation script to receive and artchive e-film. refer PS 10.112.20.84
3. Report Service need prepare the report archive rule and refer PS 10.112.20.84
4. Need install a virtual printer in PS server to simulat the terminal paper print works. Ref to PS 10.112.20.84
5. Add a system varable PYTHONPATHand point to folder..\ RF\_PUMA\PUMA\Libs. If not the self define library cannot identfy in robot framework.
6. All document and code file of automation should use the UTF-8 format.

Q&A：

Book： <https://item.jd.com/11993134.html>

Blog：<https://www.cnblogs.com/wanghao4023030/p/11417438.html>

Python + robot framework + ide pycharm

Study advise： parametersAndSettings –> patientlib->reportlibarary->filmlibrary->TermianlLibarary.

CodeSVN: https://shgwp1065/svn/kiosk/documents/QA Team/PUMA\_AUTO

Add 2019-11-29:

There are some prepared work need to do for new test environment.

1. Prepare a virtual machine which have condition to install the PS system. Such as .net, database, iis and other configurations.
2. Install the PDF creator software, we suggest to install lastest version. It can store the PDF file with defferent folders to enhance the performance issues. The virtual PDFCreator printer should be the default printer. And the printer should save the files to local disk.
3. Import the OCRConfigDB data to init the settings for test script. Notice the DB structure change that we will need change the DB script.



1. Import the printer, terminal, parameter realted database table data to init the test environment. Notice the DB structure change that we will need change the DB script.



1. Copy the report OCR service config file with prepared one. Which rule is for the test script to recognize the PDF report and archived in the PS system. Copy and cover the file to PS server and path is “C:\Program Files\GX Platform\GXWeb\Site\service\ReportService\”

****

1. Copy the folder to test PS system which will used for other cases such as notify report archived. The folder name is ” PUMA\_AUTO\_Ref” and athived at ..\\ QA Team\PUMA\_AUTO