SELECTING PERFORMANCE TEST TOOLING – PART 4

Decision making time

Decision making process

Considering the fact that I am not too fond of Sikuli and SilkTest is disqualified because it cannot deal with the remote application the decision was tough and yet simple. I have an immediate need which needs fulfilling and besides that a customer wish to look ahead and assume we will be working on test automation in the (near) future for regression testing and end-to-end testing.

The choice was made to not go for the, very affordable, commercial tool at this point in time, but rather go the open source road. Sikuli it is.

Experiences with Sikuli

As stated above Sikuli was not my preferred tool, since it is heavily depending on screen captures, however when I was finally working with it I started to get a liking for it. It has grown on me by now. Scripting in it can be both difficult and extremely easy.

I decided to approach the functional measuring with SIkuli as a pure testautomation project, but then a bit less reusable since it depends on screenshots. Starting where I generally start; starting the application and logging in to the system, was simple enough. Although still not exactly intuitive. The startup code looks something like this:

```
cmd = 'mstsc.exe "C:\\Program Files\\RemotePackages\\AppAccNew.rdp"'
def startApp():
    from time import time
    startTime = time()
    Log.elog('Startingeptatie RDP Session')
    App.open(cmd)
```

```
def CheckSelected(img1):
    hover(img1)
    areaSel = getLastMatch()
    #print areaSel
    click(img1)
    if not areaSel.right().wait( 90 ):
        Log.elog("Selecting seems to have failed")
        return True
    else:
        return False


def CloseProfit():
    if exists( HRM ):
        VerifyHRM()
    else:
        exit()
    keyDown(Key.ALT)
    keyDown(Key.F4)
    keyUp(Key.ALT)
    keyUp(Key.F4)
    wait(0.5)
```

On top of a separate, reusable login (and logoff and shutdown) routine, I also built up a nice set of helpful methods for measuring time between an action and the result, verifying that the expected area indeed is selected and quite a few others. These look a bit more odd in my eyes due to the screen captures inline in the code, as you can see here.

The moment the basic functions were there, e.g. click on something and expect some result with a timer in between, the rest was fairly straight forward. We now have a bunch of functional tests which, instead of doing a functional verification are focussed on duration of the calls, but for the rest it is not very far from actual functional automation through Sikuli.

Conclusion

All in all it took some getting used to the fact that there is script combined with screenshots, but now that it is fully up and running the scripting is fast and easy to do. I am quite impressed with what Sikuli can do.

Source : http://martijndevrieze.net/category/load-performance-testing-2/page/2/