PAPRYQARZ
WE TEST WITH TASTE

ISTQB ma znaczenie - testowanie protokołu binarnego

wersja rozszerzona

Tadeusz Zdanowicz, Mobica

PAPRYQARZ

Robot framework:

zanim zobaczysz automatyzację:

```
$ pybot -d reports <path-to-robot-test-cases>

==============================================================================
Sample Basic Project Test
==============================================================================
Sample Basic Project Test.AllTests
==============================================================================
Should-be-able-to-login                                               | PASS |
------------------------------------------------------------------------------
Sample Basic Project Test.AllTests                                    | PASS |
1 critical test, 1 passed, 0 failed
1 test total, 1 passed, 0 failed
==============================================================================
Sample Basic Project Test                                             | PASS |
1 critical test, 1 passed, 0 failed
1 test total, 1 passed, 0 failed
==============================================================================
Output:  <current-dir>/reports/output.xml
Report:  <current-dir>/reports/report.html
Log:     <current-dir>/reports/log.html
```

Zacznijmy od początku:

Docelowe zadanie:

testy modułu przetwarzającego protokół binarny zintegrowane z robot framework

Stan początkowy:

dostępna dokumentacja protokołu i

przykładowe dane zapisane w pliku

Moduł, zdefiniowane:

- API do komunikacji z resztą systemu
- format danych oczekiwany przez resztę systemu
- API do pobierania strumienia danych

Deweloperzy!

- Zespół 3 osobowy
- Implementacja w C++
- IDE: QT Creator
- Poza zdefiniowanymi API pełna dowolność implementacji.

Testerzy i ich narzędzia!

- 2 testerów
- skrypty w Pythonie
- IDE: Geany + pluginy
- wymaganie integracji z systemem testów klienta
- konieczność stworzenia i zaakceptowania planu testów

Wieloplatformowość

- Docelowy moduł na różne systemy:
  windows/linux/ios, 32/64 bity
- Użyty system operacyjny:
  Ubuntu LTS lub bazująca na nim wersja Minta

Linia komend?

Tak! Po pewnym czasie okazuje się, że narzędzia linuxowe i skrypty bash potrafią oszczędzić sporo pracy!

np. zamiast dodatkowej funkcji w pythonie:

```
for plik in *.dat; do python mojskrypt.py $plik; done
```

Protokół:

- dane nadawane w powtarzającym się **cyklu**, jest znacznik startu cyklu
- strumień podzielony na **ramki** o wielkości zależnej od typu np. pogoda, czas, ogłoszenia
- ramki z pól bitowych np. 17 bitów

bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
field	type				variant			data							code				

HELP?

ISTQB podpowiada:

- planowanie testów -> wyrocznia?
- analiza i projektowanie testów -> dane?
- testowanie -> narzędzia?
- wyniki testów -> integracja?
- dokumentacja -> tak, dokumentacja!

Planowanie:

Z dokumentacji:

- jakie ramki nas interesują
- wartości brzegowe
- przypadki negatywne

Konsultacje z klientem:

- dostęp do infrastruktury
- integracja

Dane!

Na początku pętla:

-> brak narzędzi do weryfikacji danych

-> brak zweryfikowanych danych żeby sprawdzić narzędzia

Co teraz?

Rozwiązanie:

Manualnie tworzenie danych:

- Bless hex editor
- Ghex

Zaleta: 100% kontroli nad zawartością i oczekiwanymi wynikami

Wada: długi czas tworzenia (docs)

Po przełamaniu impasu:

dzięki danym implementacja skryptów:

- sprawdzanie danych
- tworzenie danych
- manipulacja danymi

```
0101001100100000 0100        0              1  11011 011       0101100010010100 00011   111101  0
21280        4       0             1  27  3      45348 (1983.01.14)3       61      0       7
5320         4       0             1  1b  3      b124          3       3d      0       7
5320 476D 6248 3F47
Save to output?(q for quit) [n]
```

Krok po kroku:

- zaczynając od pojedynczych ramek
- poprzez ciągi
- do pełnego cyklu
- także dane nieprawidłowe

Manipulacja danymi -> niepotrzebny wysiłek?
NIE -> testy wydajnościowe

PAPRYQARZ

Spotkanie z Pareto:

najwięcej problemów:

- przesunięcia bitowe -> dane!
- wyświetlanie wyników-> specjalizowane funkcje

```
f decode_frame(self, binary):
    frame_code = binary[0]
    frame_type = binary[1]>>12
    frame_variant = (binary[1]>>11) & 1
    part = (binary[1]>>5) & 31
    self.update_display_strings("PART", part, "TYPE", frame_type,       \
    "CODE", frame_code, "VARIANT", frame_variant)
```

Kombinacja:

- dane do których mamy zaufanie
- sprawdzone skrypty

Można podłączać do Robota!

- NIE skrypty py -> narzędzie kompilowane po każdej zmianie w kodzie, używające funkcji testowanego modułu

Dokumentacja!

- Tak, to boli!
- Tak, jest konieczne

Potwierdzone praktyką prawo Eaglesona:

“Twój kod do którego wracasz po ponad pół roku jest jak kod obcej osoby”

Dokumentacja egoistycznie

- postawiony cel testów
- plan testów
- implementacja
- tworzenie i manipulacja danymi:
  - użycie narzędzi z przykładami
  - krok po kroku każdej procedury
  - możliwe usprawnienia
  - “nice to have”

PAPRYQARZ

Rezultat końcowy:

zadowolony klient

zadowoleni programiści

zadowoleni testerzy

(chwilowo :D)

Uwagi o ISTQB na koniec

- w codziennej praktyce nie używam ISTQB?
- a jednak! ISTQB to usystematyzowana wiedza praktyczna
- procedury/zalecenia/informacje nie są wzięte z sufitu ale z praktyki
- sylabus jest aktualizowany!

czas na Q&A!

ISTQB a wartość dodana w projekcie?

Moim zdaniem TAK! Dzięki ISTQB powstały narzędzia manipulacji danych ułatwiające późniejsze zmiany testów. Idąc na łatwiznę następna osoba zmieniająca testy musiała by to robić w edytorze hexadecymanlnym!

Dla mnie osobiście plusem było rozwinięcie znajomości Pythona , Linux.

ISTQB Advance - czy warto?

Planuje dopiero podejść do egzaminu więc powiem tak: moim zdaniem powinien być przydatny w dużych projektach robionych przez duże firmy/korporacje - tam gdzie wiedza z ‘Foundation’ przestaje wystarczać.

Czy tester z ISTQB jest lepszy od testera bez?

Dla mnie liczy się czy dobre podejście i pasja do testowania. Nawet bez certyfikatu można zapoznawać się z wiedzą i zdobywać informacje! Posiadanie certyfikatu -> zdanie egzaminu -> zapoznanie się z sylabusem.

Nie zwalnia to jednak od myślenia i doskonalenia umiejętności ;)