Тестирање са потпуном покривеношћу није довољно

Март 2018

1 Тестирање и потпуна покривеност

Покривеност кода је корисна метрика тестирања, али, висока покривеност није *циљ*. Код може да има 100% покривености, па да ипак има буба. Рецимо, ако неки код *недостаје*, онда програм не ради оно што би требало да ради.

1.1 100% покривен код са бубом

У модулу који је имао 100% покривености, како редова кода, тако и грана, постојала је прилично озбиљна буба.

Модул се бавио листама временских контрола (ВК), у виду (двоструко) повезаних листи. Сваки члан носи податак "колико тикова треба да прође од истека претходног члана док овај не истекне". На пример, листа би могла да изгледа овако у неком тренутку (ради прегледности, само показивачи "у смеру унапред" су приказани):

```
T1(100) -> T2(20) -> T3(38) -> T4(2)
```

где видимо да су активне четири ВК, са временима истицања:

- T1 за 100 тикова
- T2 за 120 тикова
- T3 за 158 тикова
- T4 за 160 тикова

Ово чини обраду листа ВК прилично ефикасном. "Тик" је јединица мере времена у датој примени (ms је честа).

Буба је проста - ако укидамо ВК, дакле изланчавамо је из листе, а она је *прва* у листи (следећа да истекне), то није обрађивано исправно. Време док та ВК истекне *није* додавано на време за следећу ВК.

Ради илустрације, претпоставимо да има две ВК, обе покренуте да трају 100 тикова, али, са једним тиком размака. Дакле, по покретању друге ВК, листа је изгледала овако:

```
T1(99) -> T2(1)
```

Након што је прошло неких 40 тикова, листа би била:

```
T1(59) -> T2(1)
```

Онда је T1 укинута, па би листа *требала* да буде:

```
T2(60)
```

Али, због бубе, листа је *у ствари* била:

```
T2(1)
```

Односно, T2 ће да истекне након 41 тика, уместо након 100 тикова.

Постојала је грана кода која изричито обрађује овај случај. Али, код који додаје "време до истека" просто није постојао у тој грани. Ради се о Ц-у, а ред кода који је недостајао је био:

```
list->timeout_left_ms += to_remove->timeout_left_ms;
```

1.2 Зашто ("потпуни") тестови ово нису ухватили?

Тестови *јесу* тестирали избацивање првог члана листе. Али, пошто је `timeout_left_ms` унутрашња ствар модула, тестови то нису проверавали. Такође, нису проверавали ни "шта ако прође неко време након избацивања". У нашем случају, ако после избацивања прође 40 тикова, T2 *не* треба да истекне (али у ствари јесте истицао).

Очигледно, овај прост сценарио није уочен при тестирању, него је свака могућност модула независно тестирана. Тестови за укидање су тестирали само изланчавање. Тестови за "пролазак времена" су тестирали само време. Бубе се врло често крију у сценаријима који обухватају више могућности.

1.3 Наравоучење

За високо квалитетне тестове, треба увидети сценарије упоребе и тестирати их (све, по могућству). Управо *не* треба размишљати о покривености кода. Штавише, висока покривеност *не* треба да буде циљ. Не желимо да "покријемо" код, већ желимо да га *тестирамо*. Мера покривености је само врста "назнаке" - ако је покривеност ниска, вероватно тестови нису баш најбољи.