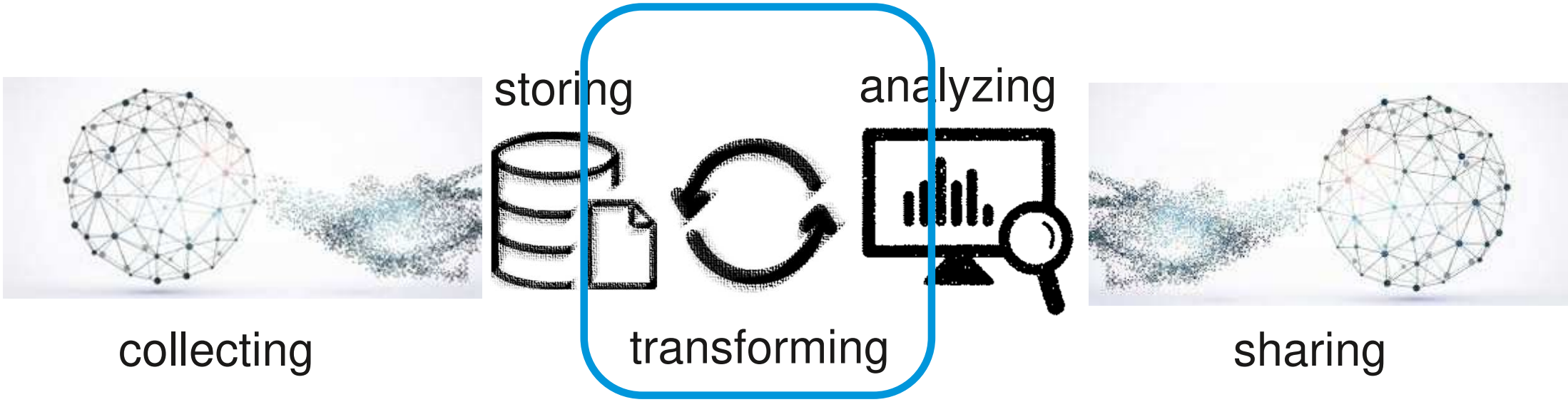
Testing data – golden vs. generic expected

Michał Pilarski

September 28, 2022

TECH6A

Data Lifecycle



Testing Data - definitions

Requirements (Reqs) - logic

convert from … to…
reformat to …
calculate based on …
etc.

Test Set

input set, according to **requirements**
covers all possibilities (in perfect world)

Actual

result collected from dev system based on **reqs** and performed on **test set**

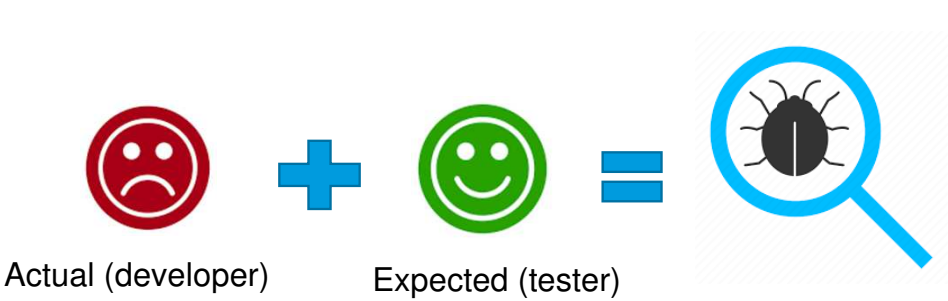
Expected

result prepared by tester based on **reqs** and linked with **test set**

Actual vs. Expected

"… expected is usually compared with actual result, and if the actual result differs from the expected one, the difference is documented and called a bug".

Reqs and Actual vs. Expected



Testing Data – can we do it manually?

Reqs

```
source = 'LETTER'
target = 'CODE'
```

```
if input == 'A':
    expected = '1'
elif input == 'B':
    expected = '2'
elif input == 'C':
    expected = '3'
elif input == 'D':
    expected = '4'
else:
    expected = 'NULL'
```

Test Set

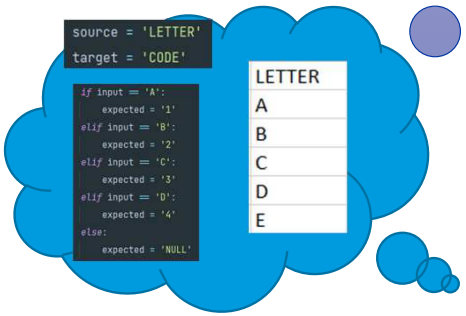
LETTER
A
B
C
D
E

Dev System

Actual

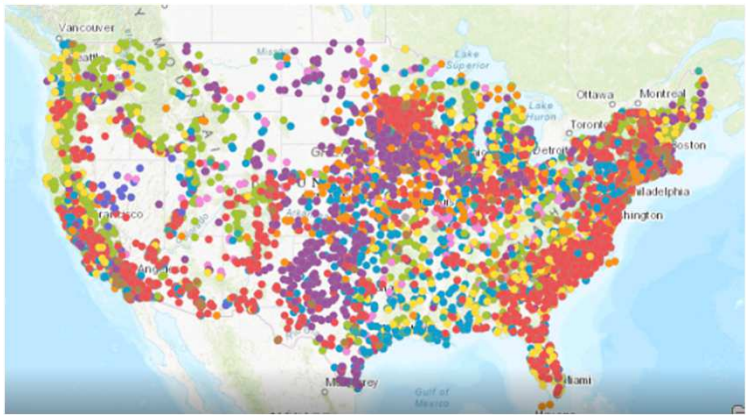
CODE
1
2
3
4
NULL

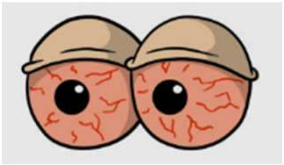
Test Assertion



Expected

1, 2, 3, 4, NULL





Testing Data (Automatically) – golden expected

Reqs

```
source = 'LETTER'
target = 'CODE'
```

```
if input == 'A':
    expected = '1'
elif input == 'B':
    expected = '2'
elif input == 'C':
    expected = '3'
elif input == 'D':
    expected = '4'
else:
    expected = 'NULL'
```

Test Set

LETTER
A
B
C
D
E

Dev System

Actual

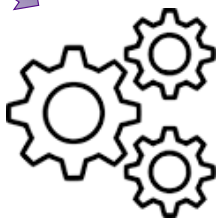
CODE
1
2
3
4
NULL

Copy
Review
Correct

Manually create

Expected

CODE
1
2
3
4
NULL



Test Assertion

Testing Data (Automatically) – golden expected - maintaining

Reqs

```
source = 'LETTER'
target = 'CODE'
```

```
if input == 'A':
    expected = '1'
elif input == 'B':
    expected = '2'
elif input == 'C':
    expected = '3'
elif input == 'D':
    expected = '4'
else:
    expected = 'NULL'
```

Dev System

```
elif input == 'Z':
    expected = '0'
```

Actual

CODE
1
2
3
4
NULL
0

Copy
Review
Correct

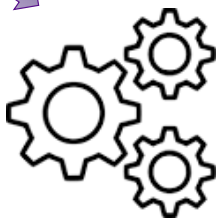
Test Set

LETTER
A
B
C
D
E
Z

Manually create

Expected

CODE
1
2
3
4
NULL
0



Test Assertion

Testing Data (Automatically) – generic expected

Reqs

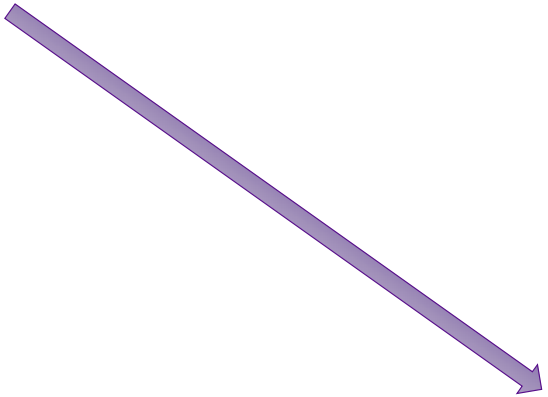
```
source = 'LETTER'
target = 'CODE'
```

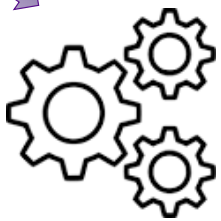
```
if input == 'A':
    expected = '1'
elif input == 'B':
    expected = '2'
elif input == 'C':
    expected = '3'
elif input == 'D':
    expected = '4'
else:
    expected = 'NULL'
```

Dev System

Actual

CODE
1
2
3
4
NULL



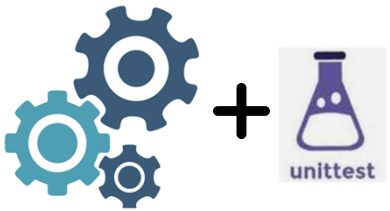


Test Assertion

Develop

Test Set

LETTER
A
B
C
D
E



Test System

Expected

CODE
1
2
3
4
NULL

Testing Data (Automatically) – generic expected - maintaining

Reqs

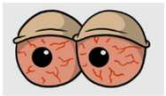
```
source = 'LETTER'
target = 'CODE'
```

```
if input == 'A':
    expected = '1'
elif input == 'B':
    expected = '2'
elif input == 'C':
    expected = '3'
elif input == 'D':
    expected = '4'
else:
    expected = 'NULL'
```

Dev System

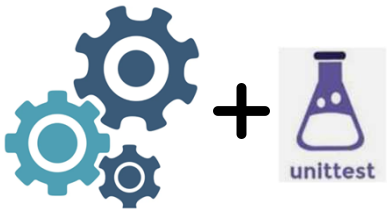
```
elif input == 'Z':
    expected = '0'
```

Develop



Test Set

LETTER
A
B
C
D
E
Z



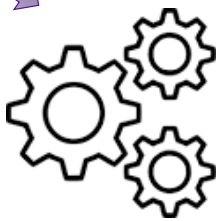
Test System

Actual

CODE
1
2
3
4
NULL
0

Expected

CODE
1
2
3
4
NULL
0



Test Assertion

Testing Data (Automatically) – golden vs. generic expected

Golden Expected

Prepared expected data set via:

1. Develop system (review&correction needed)
2. Manually (:/)

+ easy and fast to prepare
- hard to maintain
- dependent on dev sys
- not fully reliable

Generic Expected

Prepared expected data set via test system

- time consuming to prepare
+ easy to maintain
+ totally independent
+ fully reliable (unit tests)

Thank you!