

Name: Hasan Yahya
 Roll-no: 22L-7971

Prior To Training
 Quiz - 4

You are given 7 constraints written in OCL. Each contains a specific code smell. Your task is to identify the bad smell and give a solution for it.

- 1) If an order exists, and its price is above 1000, and it's pending, then the customer must have logged in within the last 24 hours.

context Order

inv:

self.price > 1000 implies

self.status = 'Pending' implies

self.customer.lastLogin >= Date.now() - 1

Implies Chain

context Order

inv:

(self.price > 1000 and self.status = "Pending")
 implies self.customer.lastLogin ≥ Date.now() - 1

4

- 2) Each customer's orders must all have valid details and those details must have a non-empty shipping address.

context Customer

inv:

self.orders->forAll(o | o.details->forAll(d |
 d.shippingAddress <> ""))

For All Chain

3

~~context Customer~~
~~self.orders~~ ~~forAll~~ ~~o~~ ~~o.details~~ ~~forAll~~ ~~d~~ ~~d.shippingAddress~~ ~~<>~~ ~~""~~ ~~)~~
 context Customer
 self.orders->forAll(o | o.details->forAll(d | d.shippingAddress <> ""))

- 3) If a cart has products, the user must not be a guest.

context ShoppingCart

inv:

if self.productId->size() > 0 then

self.customer.user.email <> 'guest@example.com'

else true endif

Verbose Expression

context ShoppingCart

inv:

self.productId->size() > 0
 implies self.customer.user.email <> "guest@example.com"

4

- 4) All orders must have a status and a price defined, and no order should be in 'Canceled' state.

context Order

inv:

self.status \neq "" and self.price > 0 and self.status \neq 'Canceled'

Duplication

context Order
inv: self.price > 0 and self.status \neq "Cancelled".

3

- 5) If a user is a customer, they must have a billing address.

context User

inv:

self.isInstanceOf(Customer) implies self.isInstanceOf(Customer).billingAddress \neq ""

DownCasting

context Customer

inv:

self.billingAddress \neq ""

4

- 6) If the shipping type is 'Express', then shipping cost should be over 20.

context OrderDetails

inv:

if self.shippingType = 'Express' then self.shippingCost > 20
else true endif

Verbose Expression

context OrderDetails
inv: self.shippingType = "Express" implies self.shippingCost > 20.

4

- 7) Every Order's customer's default shipping address must be in Pakistan.

context Order

inv:

self.customer.defaultShipping.region.country.code = 'PK'

Long Journey

context Customer

self.defaultShipping.region.country.code = 'PK'

3

There is no need to check order,
because all customers have
default shipping address

Name: Hasan Yalçın
Roll-no: 22L-7A78

Roll-no: 22L-7978

Post Training Quiz - 4

You are given 7 constraints written in OCL. Each contains a specific code smell. Your task is to identify the bad smell and give a solution for it.

- 1) If an order exists, and its price is above 1000, and it's pending, then the customer must have logged in within the last 24 hours.

context Order

inv:

self.price > 1000 implies

self.status = 'Pending' implies

```
self.customer.lastLogin >= Date.now() - 1
```

context	Order
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15
16	16
17	17
18	18
19	19
20	20
21	21
22	22
23	23
24	24
25	25
26	26
27	27
28	28
29	29
30	30
31	31
32	32
33	33
34	34
35	35
36	36
37	37
38	38
39	39
40	40
41	41
42	42
43	43
44	44
45	45
46	46
47	47
48	48
49	49
50	50
51	51
52	52
53	53
54	54
55	55
56	56
57	57
58	58
59	59
60	60
61	61
62	62
63	63
64	64
65	65
66	66
67	67
68	68
69	69
70	70
71	71
72	72
73	73
74	74
75	75
76	76
77	77
78	78
79	79
80	80
81	81
82	82
83	83
84	84
85	85
86	86
87	87
88	88
89	89
90	90
91	91
92	92
93	93
94	94
95	95
96	96
97	97
98	98
99	99
100	100

Inv:

price > 1000 implies
status = 'Pending' implies
customer.lastLogin >= Date.now() - 1
context Order
Inv:

~~total = self.price + tax~~
~~if (self.price > 1000 and self.status = 'pending')~~
~~implies self.customer.lastLogin >= Date.now() - 1~~
~~and self.status = 'pending'~~
~~(self.price > 1000 and self.status = 'pending')~~
~~implies self.customer.lastLogin >= Date.now() - 1~~

→ (self.price > 1000 and self.status = "pending") implies self.customer.lastLogin >= Date.now() - 1

- 2) Each customer's orders must all have valid details and those details must have a non-empty shipping address.

context Customer

inv:

```
self.orders->forAll(o | o.details->forAll(d |
    d.shippingAddress <> " " ) )
```

Context Customer
inv in order

if.orders->forAll(o | o.details->forAll(d |
 d.shippingAddress <> ""))

context Customer
 invs self.orders.details → for All (d | d.shipping Address

- 3) If a cart has products, the user must not be a guest.

context ShoppingCart

inv:

if self.productId->size() > 0 then

```
self.customer.user.email <> 'guest@example.com'
```

```
else true endif
```

Verbuse Expression

Long ~~Long~~ Young

comfext shopping cart

self.product_id → ...
... CustData

implies self-customer-cost.

self. user_email

Self context Customer

~~def \vec{a} cost~~
~~def \vec{a} cost~~ Cost Data

- 4) All orders must have a status and a price defined, and no order should be in 'Canceled' state.

context Order

inv:

self.status <> "" and self.price > 0 and self.status <> 'Canceled'

context Order

inv:

let s = self.status

s <> "" and s > 0 and s <> "Cancelled"

Duplication

4

- 5) If a user is a customer, they must have a billing address.

context User

inv:

self.oclIsTypeOf(Customer) implies self.oclAsType(Customer).billingAddress <> ""

context Customer

inv:

self.billingAddress <> ""

Because customer is the only possible user type in Diagram

4

~~Type~~ ~~Downcasting~~
Downcasting

- 6) If the shipping type is 'Express', then shipping cost should be over 20.

context OrderDetails

inv:

if self.shippingType = 'Express' then self.shippingCost > 20
else true endif

context OrderDetails

inv:

self.shippingType = "Express" implies

Verbose Expression

Type Based cond

self.shippingCost > 20

4

- 7) Every Order's customer's default shipping address must be in Pakistan.

context Order

inv:

self.customer.defaultShipping.region.country.code = 'PK'

Long Journey

context Customer

self.defaultShipping.region.country.code = "PK"

let shipping = self.customer.defaultShipping

context Order

inv:

self.customer.shipping = "PK"

4