Semestrální práce z předmětu B6B36TS1

Vypracovaly: Shchepetova Sofia (102), Daria Dunina (106)
Semestr: Letní 2019/2020

1. Návrh testovací strategie

1.1. Popis funkcionality aplikace

Aplikace reprezentuje zjednodušenou implementaci blockchain platformy a nad ní je realizován systém pro sledování toku potravin od jejich vypěstování, přes jejich procesování, skladování, distribuci, až po prodej zákazníkovi. Součástí celého ekosystému jsou Entity jako Farmář, Zpracovatel, Sklad, Prodejce, Distribuce, Zákazník, které tvoří ekosystém bodů, přes které tečou potraviny. Cílem ekosystému - řetězec vzniku, zpracování a prodeje potravin - je zaručit, aby v každém bodě systému bylo možné dohledat, kterými body potravina prošla, co se s ní v každém bodu dělo a zároveň, aby nebylo možné tyto informace někým zmanipulovat.

1) Hlavní entity (*Parties*) se kterými pracujeme jsou Farmer, Processor, Storage, Distributor, Seller, Customer, Product.
2) Jednotlivé strany si v ekosystému předávají potraviny (*ProductTransaction*) a za potraviny platí peníze (*MoneyTransaction*).
3) Každá strana s potravinou provede *příslušné operace*. Každá operace má parametry - např. operace Skladování - délka 4 dny, teplota 12 stupňů, vlhkost…
4) Systém je realizován pomocí velmi zjednodušené *blockchain platformy*. Každá Transakce s parametry a identifikací party, která je provedla, je zaznamenána a odkazuje se na předchozí Transakci. Již provedené Transakce a jejich parametry nelze zpětně upravovat. Každá Party má Immutable List vlastních transakcí a transakcí provedených v celém systému.
5) V každém bodě životního cyklu potraviny lze získat *věrohodné informace* o tom, přes jaké parties potravina prošla a jaké operaci s ní provedly.
6) Systém realizuje tzv. *Kanály* (Channels). Kanál spojuje parties, operace a má také svůj vlastní řetěz událostí. V systému je vlastně 2 druhy kanálů - PaymentChannel (peněžní transakce) a SellingChannel (transakce s potravinami).
7) Systém detekuje tzv. *double spending problém*, např. kdy se zpracovatel snaží prodat tu samou potravinu dvakrát.
8) Zpracování potraviny u party je realizováno stavovým automatem - tedy potravina prochází různými stavy než může být předána další party. Mezi jednotlivými diskrétními kroky simulace může dojít pouze k jednomu přechodu mezi stavy.
9) Každá Party může zasílat požadavky do jiných Parties - "zákazník chce jablko, já to v zásobách nemám, chci u vás koupit tohle jablko".

1.2. Testovací strategie

1.2.1. Přehled částí aplikace

- Komunikace mezi jednotlivými parties– sending&gettingrequests;
- Transakce mezi parties – moneytransaction&producttransaction;
- Blockchain – přidávání transakce do immutable listů všech transakcí v systému & všech transakcí jednotlivé party;
- Detekce double spendingproblému - vSellingkanalu;

- Zpracování jednotlivých potravin pomocí stavového automatu;
- Zpracování jednotlivé potraviny v rámci jednotlivé party – výběr strategie & provedení příslušných akcí (sklad – skladování, zákazník – kupování atd.);
- Průchod jednotlivé potraviny všemi potřebnými parties.

1.2.2. Prioritizace částí aplikace

Část	**Možné poškození**	**Vysvětlení možného poškození**	**Pravděpodobnost selhání**	**Vysvětlení pravděpodobnosti selhání**
Komunikace mezi jednotlivými parties (requests)	H	Nekorektní práce celého systému, zmatek v zpracovávání potraviny	H	Velká frekvence používání; Komunikace mezi parties záleží na správném nastavení další party v systému, takže pokud klient neudělá to správně - není žádná garance korektní práce celého řetězce
Transakce mezi parties	L	Klient bude muset poslat peníze ještě jednou	M	Velká frekvence používání; Úspěšné vytvoření transakce na předání potraviny záleží na úspěšné přijetí peněz za tuhle potravinu. Pokud poslané klientem množství peněz neodpovídá ceně potraviny, žádná akce se nevykoná
Blockchain	H	Nespolehlivost celé blockchain platformy	L	Funkcionalita je řešena pomocí stabilního algoritmu s používáním Immutable listu i pomocného standardního listu
Detekce double spending problému	L	False negative/positive označení double spendingu znamená snížení důvěryhodnosti této Party	M	Velká frekvence používání; Historická chybovost kvůli závislosti průchodu simulace na původních nastaveních klienta
Zpracování jednotlivé potraviny pomocí stavového automatu	M	Zmatek v zpracovávání potraviny	L	Funkcionalita je řešena pomocí stabilního stavového automatu

Zpracování jednotlivé potraviny v rámci jednotlivé party – výběr strategie & provedení příslušných akcí	H	Nekorektní informace o toku potraviny a její nekorektní stav	L	Každá party řeší zpracování potraviny zaprvé pomocí stabilního stavového automatu, zadruhé pomocí jednoznačně určených jednoduchých funkcí, neobsahujících žádné if-else podmínky. Strategie se také vybírá podle jednoznačně určené potraviny.
Průchod jednotlivé potraviny všemi potřebnými parties	H	Nekorektní práce celého systému, zmatek v zpracovávání potraviny	H	Velká frekvence používání; Záleží na správou detekcí případné existující potraviny u příslušné party.

1.2.3. Test levels

	Pravděpodobnost selhání		
Možné poškození	**High**	**Medium**	**Low**
High	A	B	B
Medium	B	B	C
Low	C	C	C

Část	**Popis**	**Třída rizika**
Komunikace mezi jednotlivými parties (requests)	H + H	A
Transakce mezi parties	L + M	C
Blockchain	H + L	B
Detekce double spending problému	L + M	C
Zpracování jednotlivé potraviny pomocí stavového automatu	M + L	C
Zpracování jednotlivé potraviny v rámci jednotlivé party – výběr strategie & provedení příslušných akcí	H + L	B
Průchod jednotlivé potraviny všemi potřebnými parties	H + H	A

Quality characteristic	**Třída rizika**	**Test levels**

Část systému / funkce		Revize	Vývojářské testy	Systémové testy	UAT	Test v produkci
Bezchybná funkcionalita						
Komunikace mezi jednotlivými parties	A		střední	vysoká	nízká	
Transakce mezi parties	C		nízká	střední	nízká	
Blockchain	B		střední	střední	nízká	ano
Detekce double spending problému	C		nízká	střední	nízká	
Zpracování jednotlivé potraviny pomocí stavového automatu	C	ano	nízká	nízká		
Zpracování jednotlivé potraviny v rámci jednotlivé party	B	ano	nízká	střední	nízká	
Průchod jednotlivé potraviny všemi potřebnými parties	A		střední	vysoká	střední	ano
Bezpečnost						
Blockchain	B			střední		ano

2. Testovací scénáře

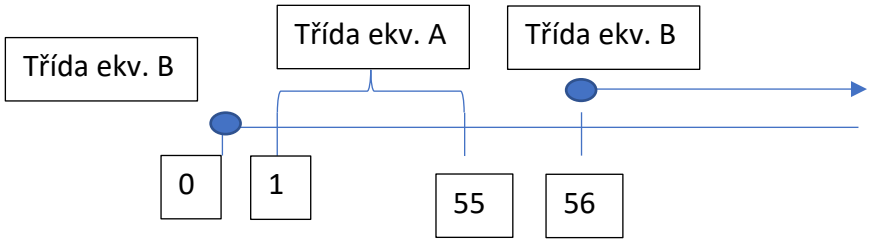
2.1.1. Testy vstupů - třídy ekvivalence

A) Main.setUpProduct()

Vstupy se rozdělí do:

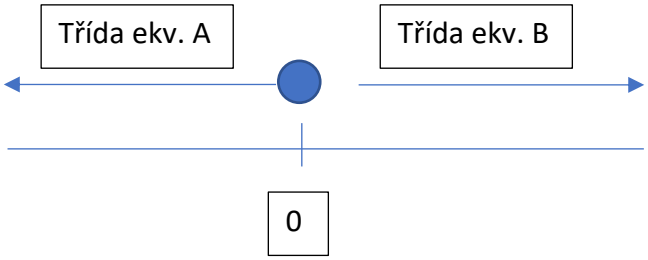
- Dvou tříd ekvivalence podle délky jmena:

Nevalidní	Validní
Délka jmena == 0 || Délka jmena >= 55	Délka jmena > 0 && Délka jmena < 55



- Dvou trid ekvivalence podle ceny produktu

Nevalidní	Validní
Cena <= 0	Cena > 0



B) Main. setUpSimulation ()

Vstupy se rozdělí do třech tříd ekvivalence:

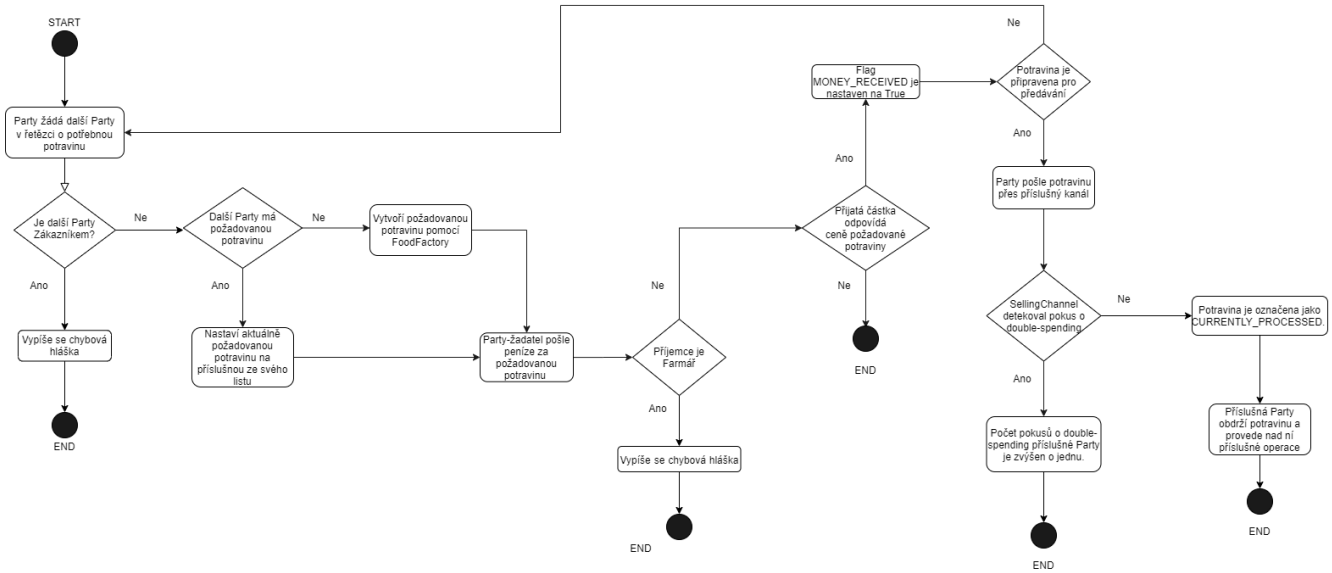
Nevalidní z technického pohledu	**Nevalidní z technického pohledu**
Částečně nastaveny FoodChain	Nelogicky nastaveny FoodChain z business pohledu: *Zákazník -> Sklad -> Prodejce -> Farmář -> Zpracovatel -> Distribuce atd.*
Validní	
Farmář -> Zpracovatel -> Sklad -> Prodejce -> Distribuce -> Zákazník	

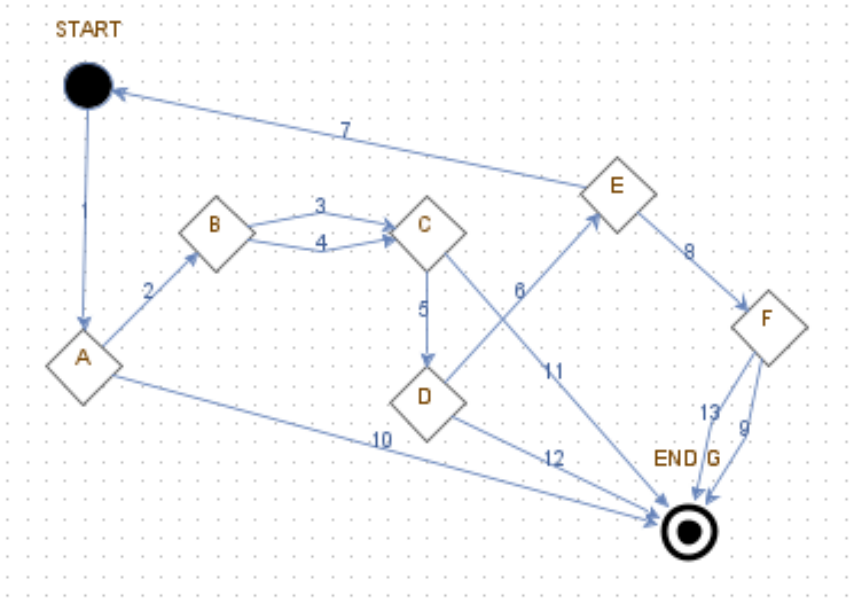
2.1.2. Testy vstupů - kombinace dat

	Nastavení FoodChain	Délka jména	Cena
1	Validní	Délka == 0 || Délka > 55	Cena > 0
2	Validní	0 < Délka <= 55	Cena < 0
3	Nevalidní z business pohledu	0 < Délka <= 55	Cena > 0
4	Nevalidní z business pohledu	Délka == 0 || Délka > 55	Cena < 0
5	Nevalidní z technického pohledu	Délka == 0 || Délka > 55	Cena < 0
6	Nevalidní z technického pohledu	Délka == 0 || Délka > 55	Cena > 0
7	Nevalidní z technického pohledu	0 < Délka <= 55	Cena > 0

2.1.3. Testy průchodů

A) Proces kupování jednotlivé potraviny *(ilustrováno níže a také v kupovani_potraviny_diagram.pdf)*





Testovací kombinace pro TDL=2

Bod	Sub-kombinace hran
A	1-2, 1-10
B	2-3, 2-4
C	3-5, 3-11, 4-5, 4-11
D	5-6, 5-12
E	6-7, 6-8
F	8-9, 8-13

Průchody

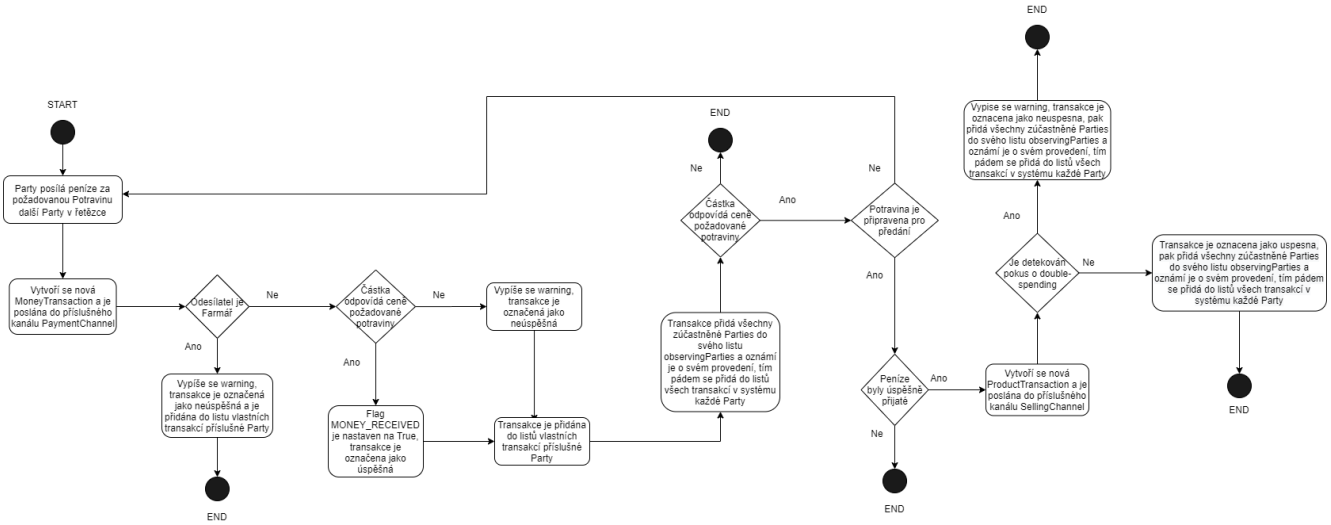
1	1-10
2	1-2-3-5-12
3	1-2-3-11
4	1-2-3-5-6-7
5	1-2-3-5-6-8-9
6	1-2-3-5-6-8-13

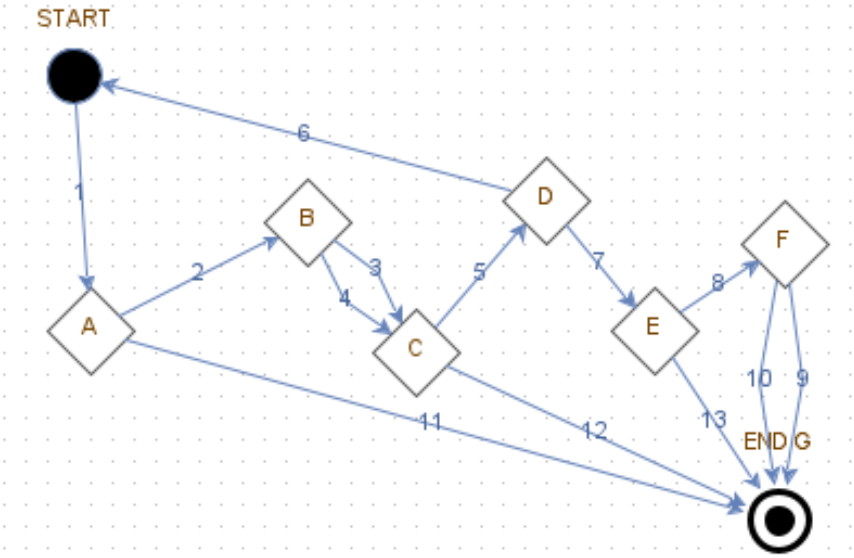
7	1-2-4-5-12
8	1-2-4-11
9	1-2-4-5-6-7
10	1-2-4-5-6-8-9
11	1-2-4-5-6-8-13

Detailní testovací scénář pro vybraný průchod

ID testu	13
Název testu	Nedostatek peněz
Průchod	1-2-3-5-12
Popis testu	Party požádala o nějakou potravinu a za ní poslala peníze, ale přijatá částka neodpovídá ceně požadované potraviny.
Vstupní podmínky	Správně nastavené parties, list existujících potravin příslušné party obsahuje požadovanou potravinu, tahle potravina má příslušný danému bodu v ekosystému stav a jsou nad ní provedeny příslušné operace. Poslána částka peněz je menší než cena potraviny.
Testovací data	Potravina - Milk Party-žadatel – Customer Party-odesilatel – Seller Poslána částka peněz - 40
Kroky scénáře	
Krok	Očekávaný výsledek
Customer žádá Sellera o Mléko pomocí metody makeRequest ()	Na instanci Sellera se zavolá metoda getRequest() a proběhne ověření, že ten request nepřišel Customeru (je poslední v řetězci a requesty by dostávat neměl)
Seller ověří, že požadovanou potravinu v zásobách má	Vypíše se hláška "Seller already has Milk" a položka currentRequestedProduct se nastaví na ten nalezený Milk. Milk je označen jako připravený pro předání Party-žadateli.
Customer pošle peníze za mléko pomocí metody makeTransaction()	Vytvoří se PaymentChannel a bude na instanci Sellera zavolána metoda receiveMoney(). PaymentChannel taky ověří, že peníze nejsou poslány Farmerem (je první v řetězci a peníze by nikomu posilat neměl).
Seller zkontroluje, jestli přijatá částka peněz odpovídá ceně Mléka	Vypíše se hláška "Not enough money!", flag MONEY_RECEIVED zůstává False a proces je ukončen.

B) Proces vytvoření transakcí *(ilustrováno níže a také v vytvoreni_transakci_diagram.pdf)*





Testovací kombinace pro TDL=2

Bod	Sub-kombinace hran
A	1-2, 1-11
B	2-3, 2-4
C	3-5, 3-12, 4-5, 4-12
D	5-6, 5-7
E	7-8, 7-13
F	8-9, 8-10

Průchody

1	1-11
2	1-2-3-5-6
3	1-2-3-5-7-13
4	1-2-3-5-7-8-9
5	1-2-3-5-7-8-10
6	1-2-3-12
7	1-2-4-5-6
8	1-2-4-5-7-13

9	1-2-4-5-7-8-9
10	1-2-4-5-7-8-10
11	1-2-4-12

Detailní testovací scénář pro vybraný průchod

ID testu	14
Název testu	Přesměrování požadavku o zaplacenou potravinu do další Party
Průchod	1-2-4-5-6
Popis testu	Party posílá další Party v řetězci peníze za požadovanou potraviny, ty jsou úspěšně přijaté, ale potravina není v zásobách Party-příjemce, takže on ten požadavek přesměruje na další Party v řetězci. Během tohoto procesu se musí vytvořit všechny potřebné Transakce.
Vstupní podmínky	Správně nastavené parties, list existujících potravin příslušné party neobsahuje požadovanou potravinu. Položka currentRequestedProduct je nastavena na požadovanou potravinu. Poslaná částka peněz odpovídá ceně potraviny.
Testovací data	Potravina - Milk Party-žadatel – Customer Party-odesilatel – Seller Poslána částka peněz - 45
Kroky scénáře	
Krok	Očekávaný výsledek
Customer posílá peníze za Mléko pomocí metody makeTransaction()	Vytvoří se nová MoneyTransaction. Je poslána do PaymentChannelu a v něm bude na instanci Sellera zavolána metoda receiveMoney(). PaymentChannel taky ověří, že peníze nejsou poslány Farmerem (je první v řetězci a peníze by nikomu posilat neměl).
Seller zkontroluje, jestli přijatá částka peněz odpovídá ceně Mléka	Vypíše se hláška "Seller has received money: 45". Flag MONEY_RECEIVED je nastaven na True. Transakce je označena jako úspěšná a je přidána do listů vlastních transakcí Sellera. Do listu transakce, který obsahuje všechny zúčastněné v současném procesu Parties, jsou přidané Customer a Seller. Transakce zavolá metodu notifyAllParties(), která má přinutit Sellera a Customera přidat provedenou transakci do listů všech transakcí v systému.
Seller ověří, že přijatá částka peněz sice odpovídá ceně mléka, ale to mléko současně nemá k dispozici	Pošle požadavek na mléko další party v řetězci a pošle za něj odpovídající částku peněz

2.1.4. Detailní testovací scénáře pro vybrané části systému

ID testu	11
Název testu	Ověření průchodu požadavku na konkrétní potravinu přes parties
Popis testu	Zákazník chci koupit nějakou potravinu, která ale zatím není v zásobách Prodejce. Ten musí poslat požadavek na tuhle potravinu do dalších parties a zaplatit za ni příslušnou cenu. Ten požadavek musí projít přes další parties až do té, která teprve tu potravinu v zásobách má. Prodej musí projit úspěšně, pokud poslána částka peněz odpovídá ceně potraviny.
Vstupní podmínky	Správně nastavené parties, list existujících potravin příslušné party obsahuje požadovanou potravinu, tahle potravina má příslušný danému bodu v ekosystému stav a jsou nad ní provedeny příslušné operace. Poslána částka peněz odpovídá ceně potraviny.
Testovací data	Potravina - Pork Party mající potravinu – Storage Poslána částka peněz - 80
Kroky scénáře	
Krok	Očekávaný výsledek
Customer žádá Sellera o Pork pomocí metody makeRequest ()	Na instanci Sellera se zavolá metoda getRequest() a proběhne ověření, že ten request nepřišel Customeru (je poslední v řetězci a requesty by dostávat neměl)
Seller zkontroluje, jestli přijatá částka peněz odpovídá ceně Porku	Vypíše se hláška "Seller has received money: 80". Flag MONEY_RECEIVED je nastaven na True.
Seller ověří, že přijatá částka peněz sice odpovídá ceně mléka, ale to mléko současně nemá k dispozici	Pošle požadavek na mléko další party v řetězci a pošle za něj odpovídající částku peněz
Stejná sekvence akce proběhne přes parties Distributor i Processor, až požadavek dorazí do Storage	Storage zkontroluje, že požadovanou potravinu Pork má ve svých zásobách a zavolá metodu sendProduct().
Zavolá se metoda makeTransaction()	Vytvoří se ProductTransaction a pošle se přes SellingChannel zpět ke Party-žadateli. List potravin Storage již to Pork neobsahuje. List jeho transakcí teď obsahuje informace o úspěšném prodeji masa.

ID testu	12
Název testu	Správná detekce double spending problému
Popis testu	Nějaká Party žádá další o nějakou potravinu. Ta další Party tu potravinu už prodávala a snaží se to

	udělat ještě jednou. Kanál na prodej potraviny musí tenhle pokus detekovat.
Vstupní podmínky	Správně nastavené parties, list existujících potravin příslušné party obsahuje požadovanou potravinu, tahle potravina má příslušný danému bodu v ekosystému stav a je jenom jedna. Položka currentRequestedProduct je nastavena na požadovanou potravinu. Položka isCurrentlyProcessed požadované potraviny je nastavena na True. Poslaná částka peněz odpovídá ceně potraviny.
Testovací data	Party která žádá o potravinu - Customer "Podvádějící" Party – Seller Potravina – Mléko Poslána částka peněz - 45
Kroky scénáře	
Krok	Očekávaný výsledek
Customer žádá Sellera o Mléko pomocí metody makeRequest ()	Na instanci Sellera se zavolá metoda getRequest() a proběhne ověření, že ten request nepřišel Customeru (je poslední v řetězci a requesty by dostávat neměl)
Seller zkontroluje, jestli přijatá částka peněz odpovídá ceně Mléka	Vypíše se hláška "Seller has received money: 45". Flag MONEY_RECEIVED je nastaven na True.
Storage ještě má v zásobách Mléko, ale formálně už bylo jednou prodáno. Ovšem se Storage snaží prodat to samé mléko ještě jednou	Seller vytvoří novou ProductTransaction obsahující výše popsanou potravinu a pošle ji do vytvořeného SellingChannelu
SellingChannel zkontroluje stav posílané potraviny a zjistí, že už byla jednou poslaná	Vypíše se hláška "ATTEMPT TO COMMIT DOUBLE SPENDING". Seller je označen jako podvádějící Party pomocí metody setDoubleSpending() a počet jeho pokusů o podvádění je kanálem zvětšen o jednu pomocí metody increaseAttempts().

3. Testování

ID testu	Název	Popis	Očekávaný výsledek	Výsledek
1	makeTransaction_MakeMoneyTransaction_ ListOfTransactionsHasCorrectInfo	Transakce se správně zaznamuje v seznamu transakcí	Správná parametry transakce	PASS
2	receiveProduct_StorageReceivesPork_ PorkParametersAreCorrect	Produkt byl uschován s dodržením	Správné parametry	PASS

		správných parametrů	uschování produktu	
3	receiveMoney_CustomerReceiveMoney_STDOUTContainsWarning			