BurgerOrderer Example

Mikael Svahnberg*

2023-02-01

Contents

1 Use Case Order Food

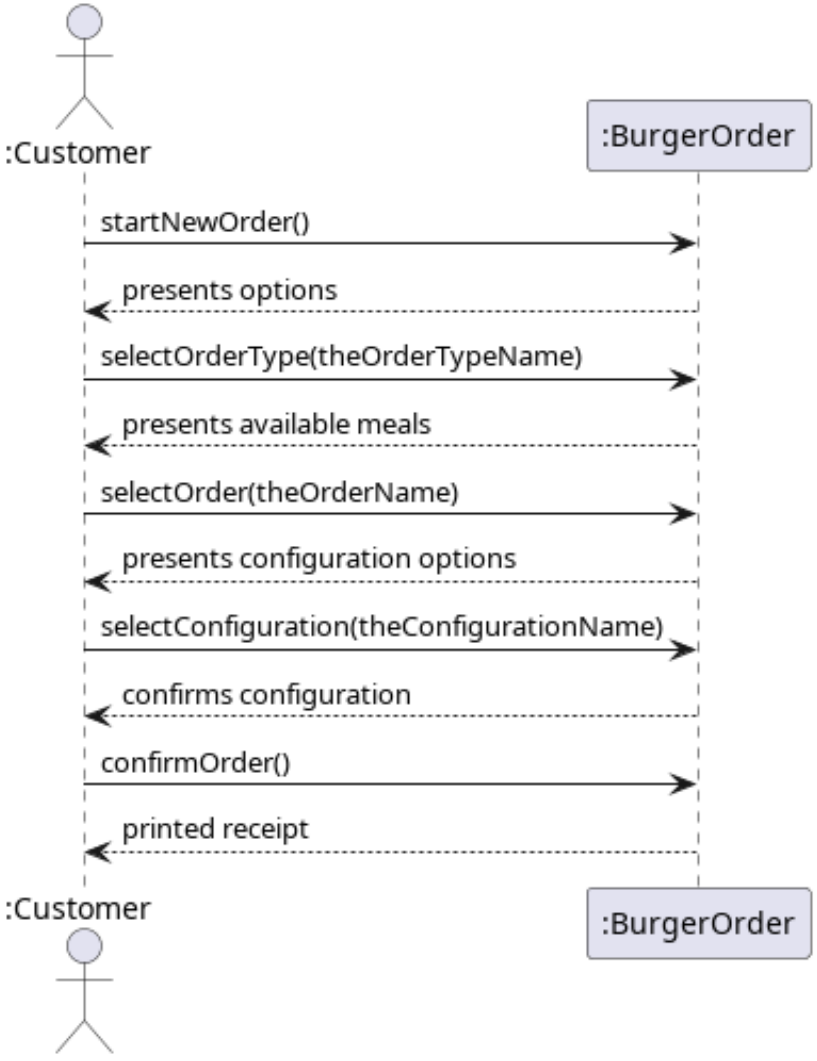
Use Case Order food
Actors Customer
Description A customer arrives at the BurgerOrderer, selects a meal, configures their burger, and orders it.
Related Use Cases Pay for ordery

Main course of events

*Mikael.Svahnberg@bth.se

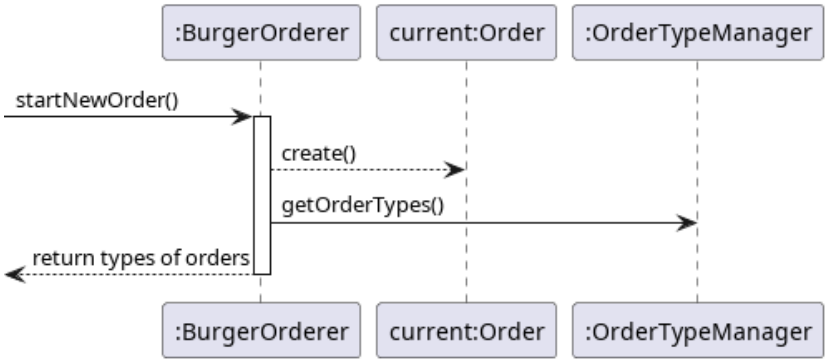
Actor	System
1. Customer arrives at BurgerOrderer and starts a new order.	
	2. System presents options [single burger, meal, dessert, dring]
3. Customer selects “meal”	
	4. System presents available meals
5. Customer selects a specific meal.	
	6. System adds the selected meal to the order. 7. System presents configuration options
8. customer selects “no onions”	
	9. System adds “no onions” to order.
10. customer selects “more bacon!”	
	11. System adds “more bacon!” to order.
12. Customer confirms order.	
	13. System initiates use case pay for order 14. System places order to kitchen and prints receipt.

2 System Sequence Diagram

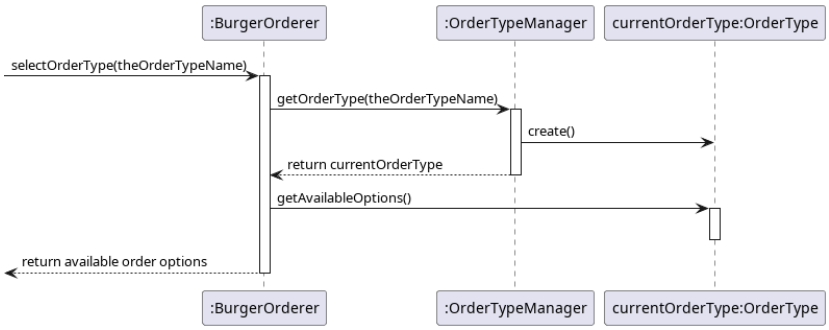


3 Interaction Diagrams (Sequence Diagrams)

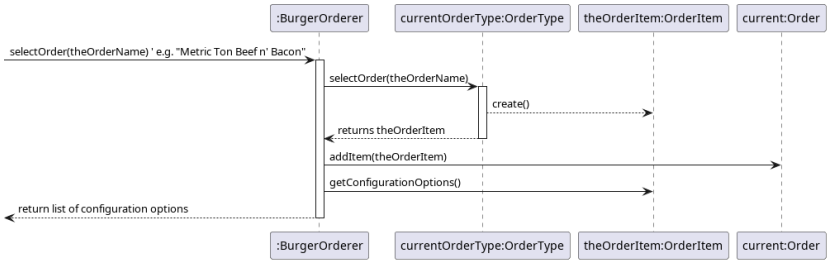
3.1 startNewOrder()



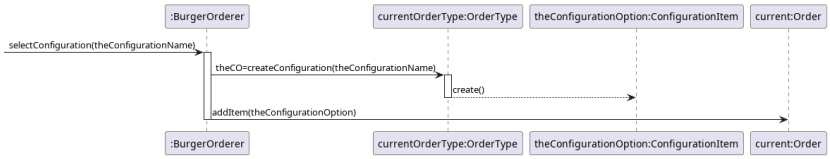
3.2 selectOrderType()



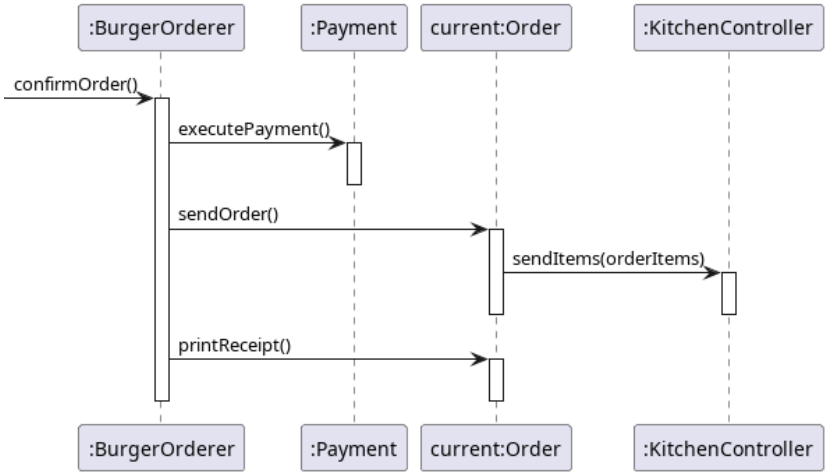
3.3 selectOrder()



3.4 selectConfiguration()

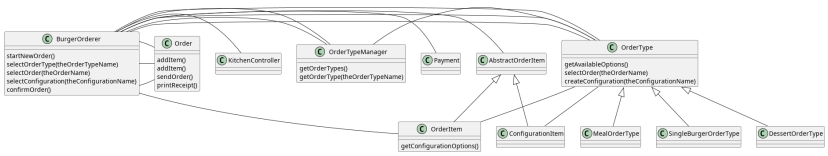


3.5 confirmOrder()



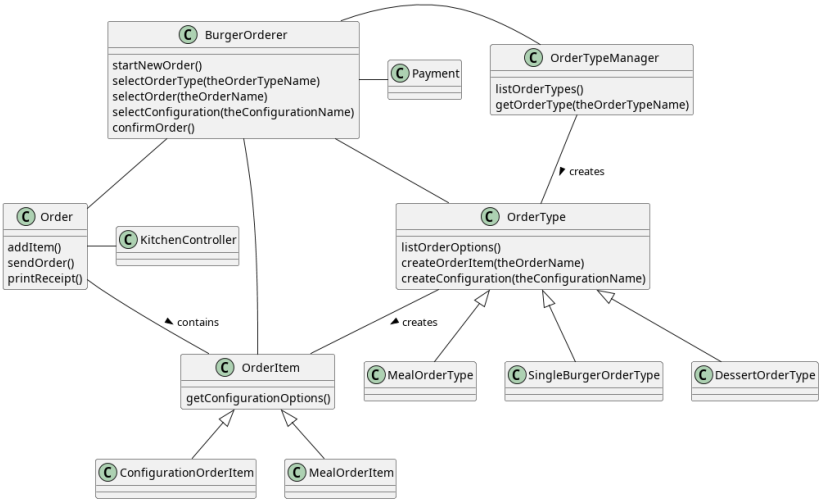
4 Class Diagram – First version

In this version, I have simply merged all of the interaction diagrams above. As is seen, this means that associations between classes are duplicated, and some associations are made to the sub-class when they should be moved up to a super-class instead. I present this as a first version, and then I will clean it up and simplify it a bit.



5 Class Diagram – Simplified

Please see the comments in the code below for information about what I have done and why,.



And there you have it. With this diagram we can now take a step back and look at a few things.

- First, `BurgerOrderer` is connected to everything! Is there anything we can do to avoid this?
- Second, the `OrderItem` inheritance hierarchy does not have that many methods currently. This *could* be because we have only modelled a single use case. But it can also indicate that maybe we do not need to have an inheritance hierarchy here. Maybe `OrderItem` with a few attributes can be sufficient.
- Third and likewise, the `OrderType` hierarchy is also suspiciously empty of methods.