Seminarium 12

Objekt-Orienterad Design, IV1350

Daniel Westerlund
daweste@kth.se

2019-04-10

Innehållsförteckning

1 Introduktion

Seminariet syfte var att lära sig att skapa ett designa ett program som kan hantera alla delar av ett försäljningsscenario (samma kravspecifikation som för seminarium 1). Designen ska ha hög ”cohesion”, låg ”coupling” samt ha en bra inkapsling utav data och ett bra designat publikt gränssnitt. Detta skall designas med hjälp av MVC och lager mönster (View kan ersättas med en klass View).

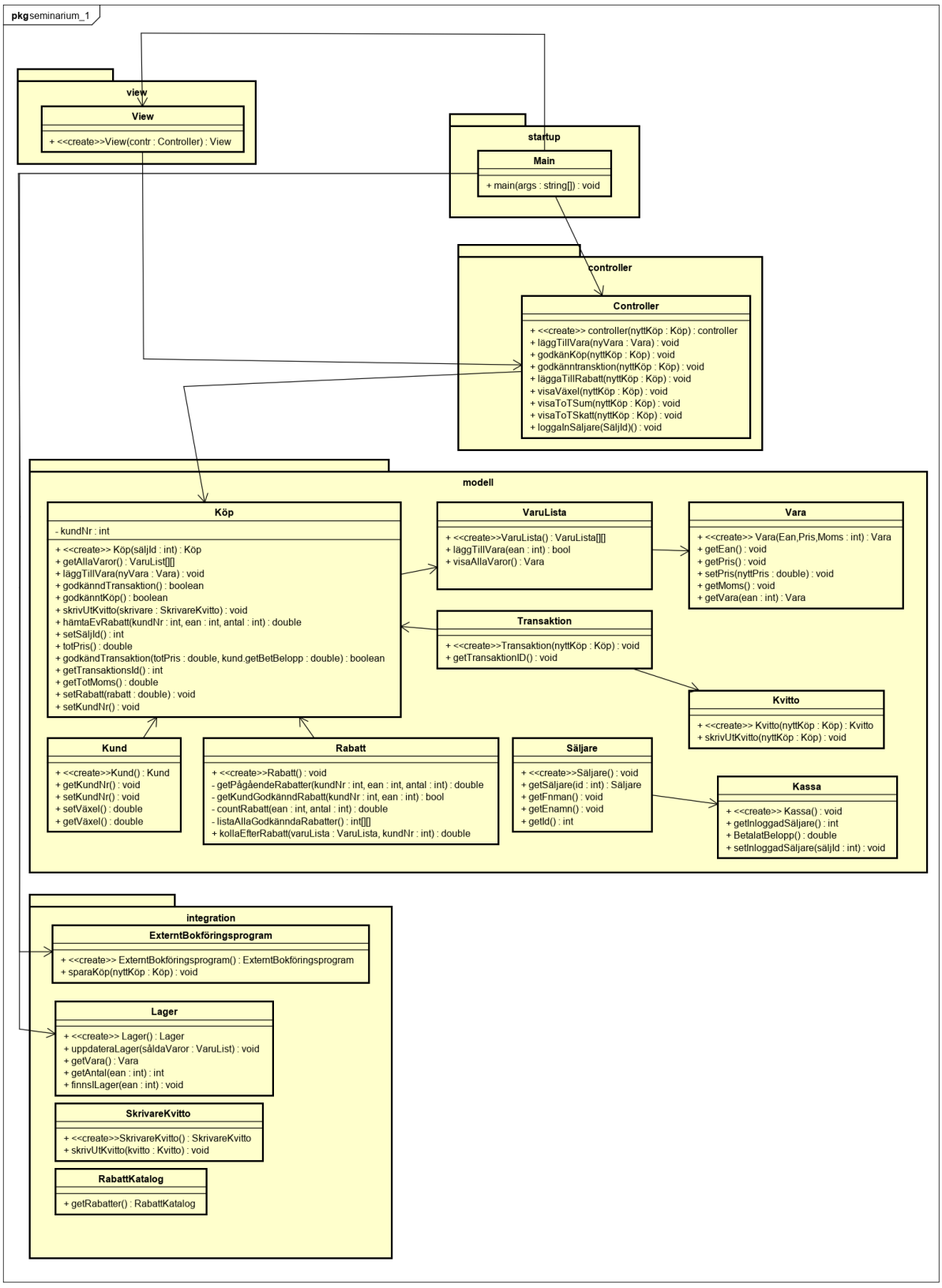
Jag har arbetat själv.

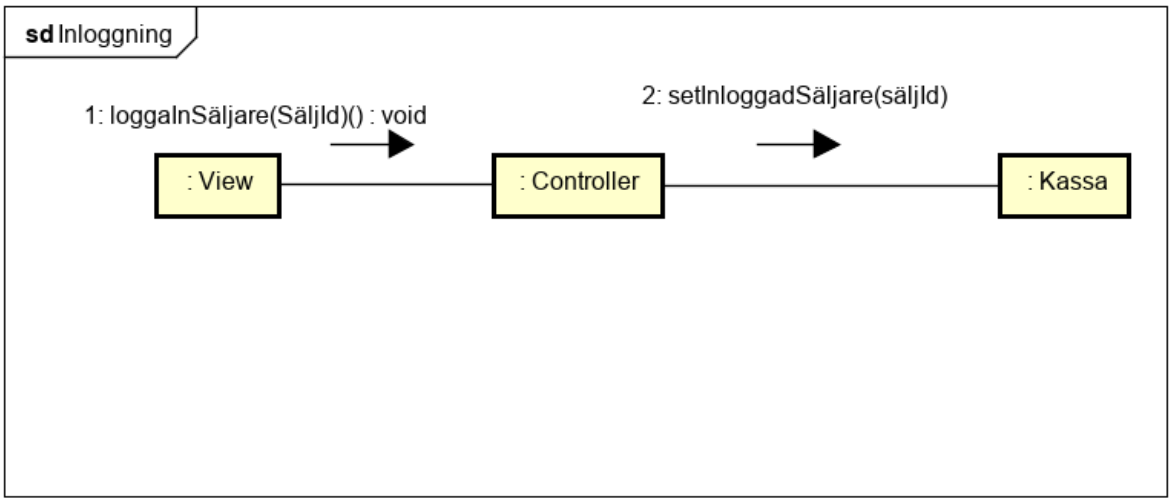
2 Metod

Efter seminarium 1 så justerades både domänmodellen och SDD. Jag började med att skissa upp en ungefärligt klassdiagram, och därefter påbörjades designen utav interaktionsdiagrammen. Använde SSD som beskriver ordningen på hur allt sker mellan säljaren och systmet, och skapade en systemoperation i taget, som t.ex. läggTIllVara, och sen tänker ett steg till, vilka funktioner/klasser behövs skapas för att kunna lägga till en vara, och därefter lags systemoperationen in den på själva kartan över alla operationer, samt adderade de metoder som kommer att behövas till klassdiagrammet. Därefter valdes nästa operation ut och samma procedur upprepas till att allt ifrån SSD och dess underliggande metoder för att lösa de anropen skapats. Kartan över alla systemoperationen trimmades genom att bara behålla stora drag av vad som händer.

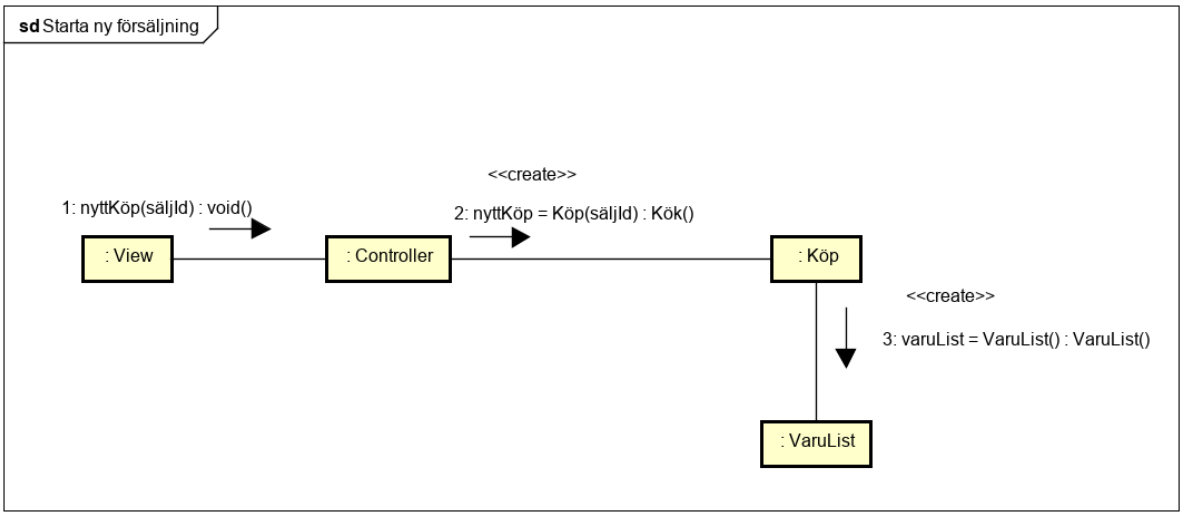
3 Resultat

Klassdiagram:

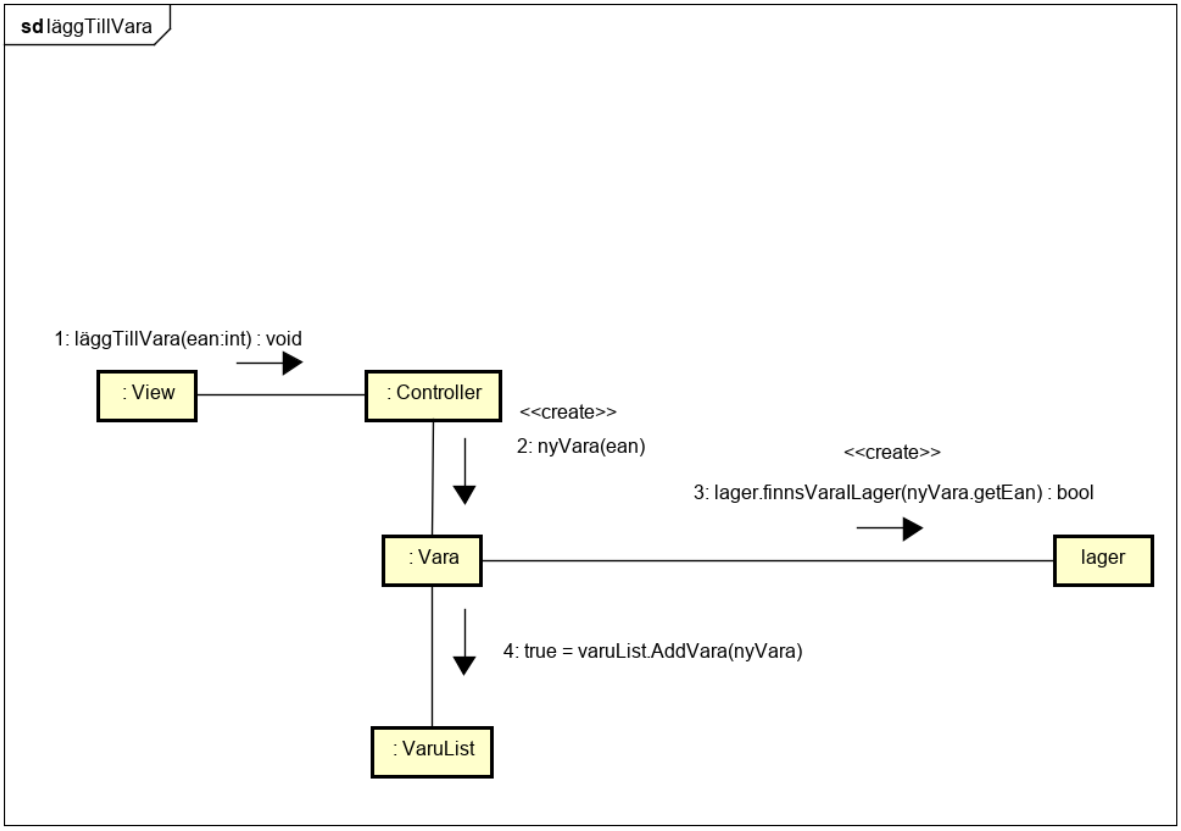




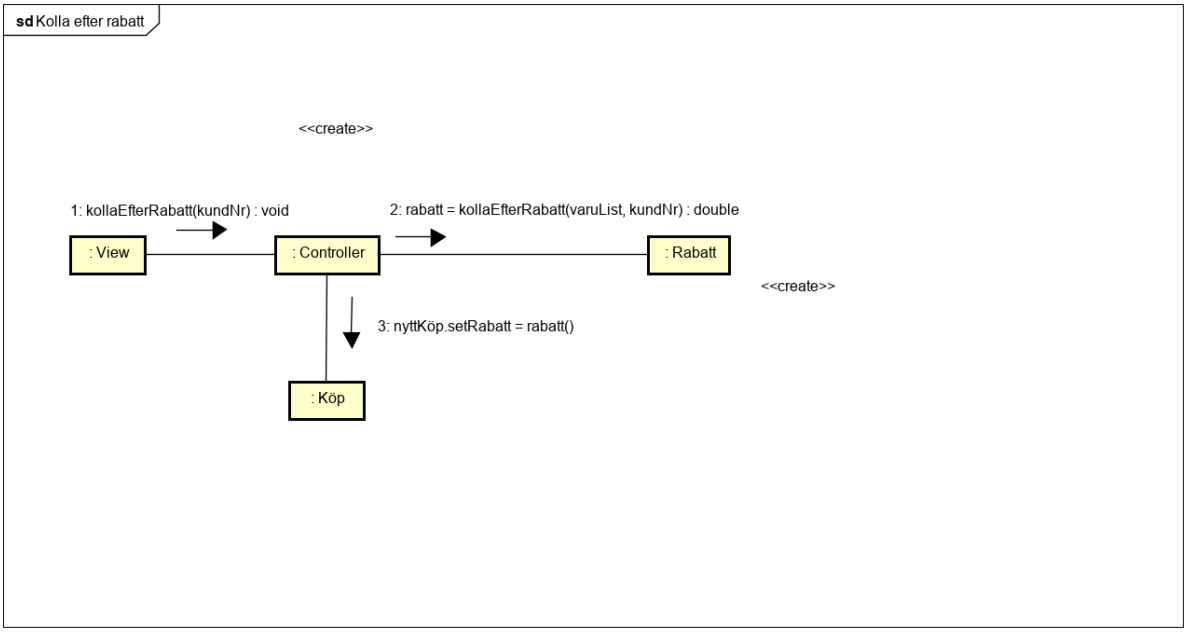
Enkel variant av en inloggning på kassa, för att veta vem som har sålt varan (visas t.ex. på kvittot).



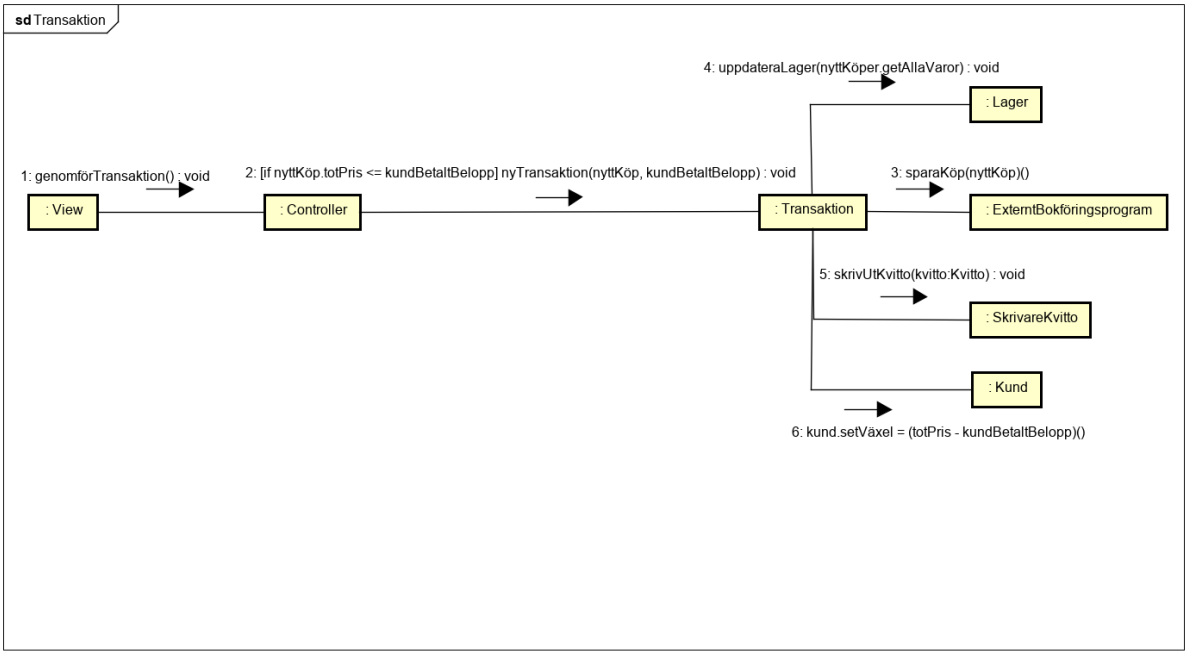
Skapar nytt objekt av klassen köp genom anrop från View argumentet inloggade säljaren, samt skapar ett objekt av VaruList, som är som en varukorg, den innehåller alla varor som kunden vil köpa.



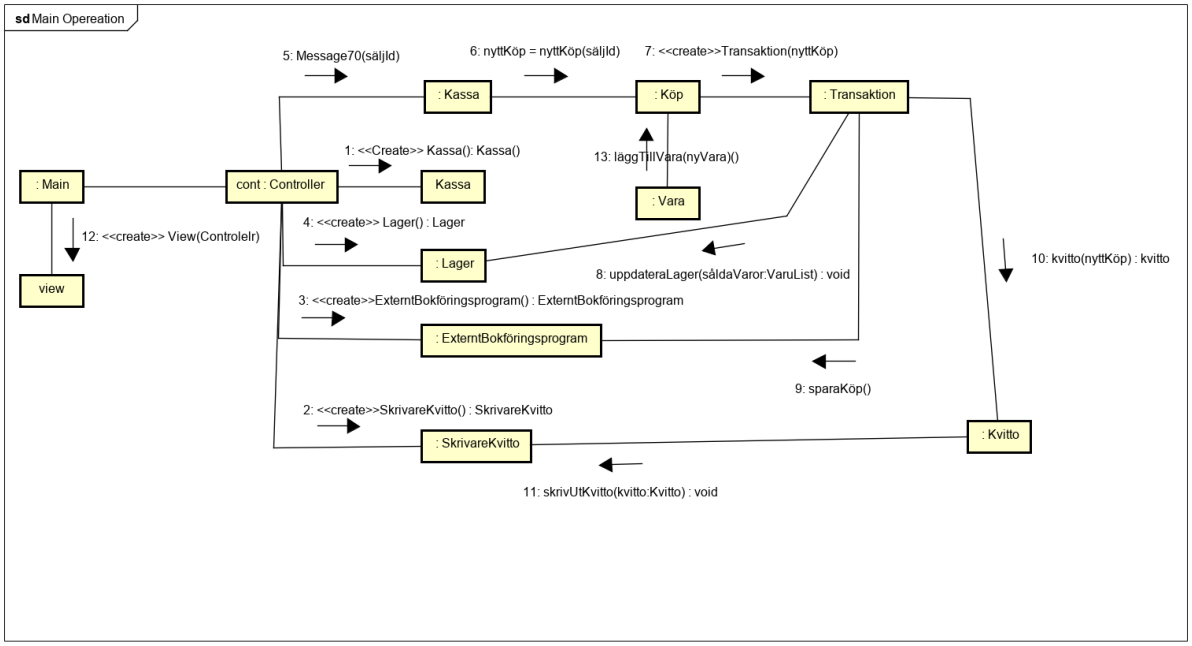
Säljaren lägger till en inskannad vara med hjälp av ean-koden, som skickas med från View till controller och sedermera skapas ett Vara-objekt som kontrolleras om det finns i lager och därefter läggs i VaruList-objeketet.



Kontrollerar om kunden har rätt till någon rabatt, uppdaterar rabatten hos köp-objektet.



Säljaren genomför transaktion, den ”går igenom” om kunden har betalat minst nyttKöp.ToTpris. Skickar vidare data till externa systemen samt visar hur mycket växel kunden ska ha tillbaka.



Översikts bild av systemoperationerna

4 Diskussion

Hade en del problem att komma igång, och veta exakt hur jag skulle lägga upp arbetet. Initialt tyckte jag att det var lättare att börja med klassdiagrammet, förmodligen för att jag har mera erfarenhet utav det. Det vart lite stökigare på det sättet, men jag ser nu varför det är rimligare att börja åt andra hållet, och nu när man har gjort det, kommer det troligtvis vara lättare att börja ”från rätt” håll. Jag upplevde att det var mycket att hålla reda på och jag försökte tänkte ett steg längre, hur kommer jag att implementera detta i programmeringen sen. Antar att detta är som med domänmodellen och SSD, att man blir bättre ju flera man skapar. Inkappslingen försökte jag lösa med att setter/getters så att när man ska ändra data på objekt ändrar man inte direkt på ”den riktiga datan”. Försökte få ner antalet publika metoder för att begränsa det publika gränssnittet, men just nu ser jag inte flera som jag kan göra om till private, men kanske upptäcker senare i programmeringsdelen, även fast det vore bra om det är löst redan nu. Det är dock lite högre ”coupling” än vad som kanske är rimligt för ett sånt här litet problem. Just nu är det ett litet problem men har man ett större så skulle detta kunna bli problematiskt rätt fort om det fortsätter så här. Även fast jag nog är mera lagd åt det så kallade ”happy-hacking” hållet så har jag insett att det är viktigt och förmodligen extremt tidsparande om man har en bra design, då kommer kodningensdelen att gå fort.

Översiktsbilden försökte jag minska så mycket som möjligt och bara ha den som en enkel överskådsbild över systemet då jag upplever att det vart enklare för mig (och förhoppningsvis andra) att tyda den, och få en bra överblick av hur systemet på fungerar på ett ungefär. Storleksmässigt tycker jag den vart bra, kanske kunde varit något större.