# Seminarium 1

Niloofar Rahmani, nrahmani@kth.se
Juni 9, 2021

## Innehåll

# 1 Introduktion

Seminarium 1 handlar för det mesta om att komma igång med att använda astah och UML genom att skapa klassdiagram, domänmodell (DM) och system sekventielldiagram (SSD). Uppgifterna i detta seminarium löstes i samarbete med Alva Ols.

Scenario för båda uppgifterna var att analysera och skapa modeller som visar en köpingsprocess i en affär. Från att kunden hämtar varorna till att kassören registrerar de och lämnar kvittot. I första uppgift var domänmodell i fokus och i andra uppgiften sekventielldiagram.

Föreläsningar och kurslitteratur har varit tillgängliga som referens och hjälpmedel.

För komplettering av detta seminarium har vi pluggat mer och försökt att få bättre förståelse. Dessutom har vi använt oss av lärarens feedback och peer review.

## 2 Metod

Lösningsprocess för uppgift 1 började med att hitta alla möjliga klasser, då togs alla namn i scenariobeskrivning (noun identification metod) och sedan användes category list från kurslitteraturen för att tänka mer kreativt och hitta ännu mer klasser. Efteråt bestämdes vilka klasser som var passande och skulle behållas. Nästa steg innebar att skapa attribut då en del av klasserna som hade hittats kunde inte vara en klass för sig utan en egenskap till en annan klass, då skrevs de som attribut. Sista steg var att skapa föreningar (associstions) för att visa hur processen går till, därför försöktes att beskrivningar skulle vara tydliga och passande. När detta var klar, undersöktes möjliga fel och namnändringar skedde.

Under komplettering tänkte vi på samma sätt med denna gång tänkte vi mer rätt och tog bort de onödiga association och klasser så att det blev lättare att förstå diagrammen.

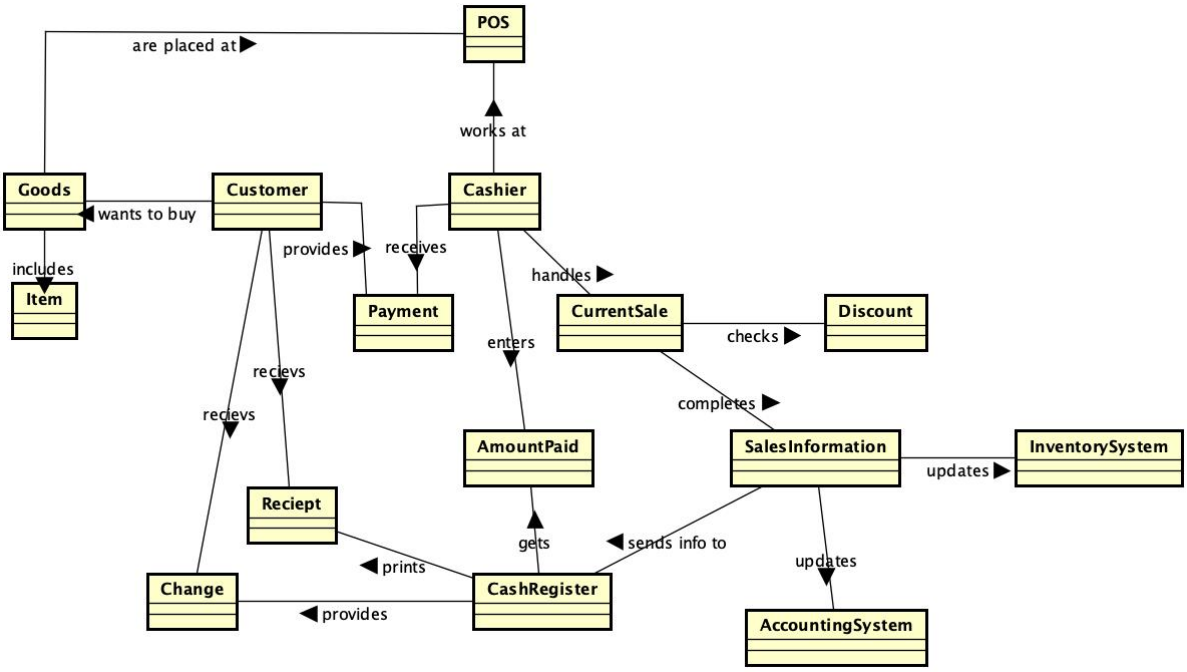
Uppgift 2 handlade om meddelanden som gick fram och tillbaks mellan de huvudklasserna som var inblandade enligt scenariot. Därför skapades först de klasserna och meddelandet mellan varje skrevs samtidigt som scenariot lästes rad för rad, olika loop och if satser skapades när det behövdes och till slut skedde ändringar för att göra diagrammet tydligare.

För både DM och SSD var vår metod att fixa de sakerna som vi hade fått feedback på och jämföra det slutliga versionen med vår gamla och vår feedback igen för att undvika samma misstag.

# 3 Resultat

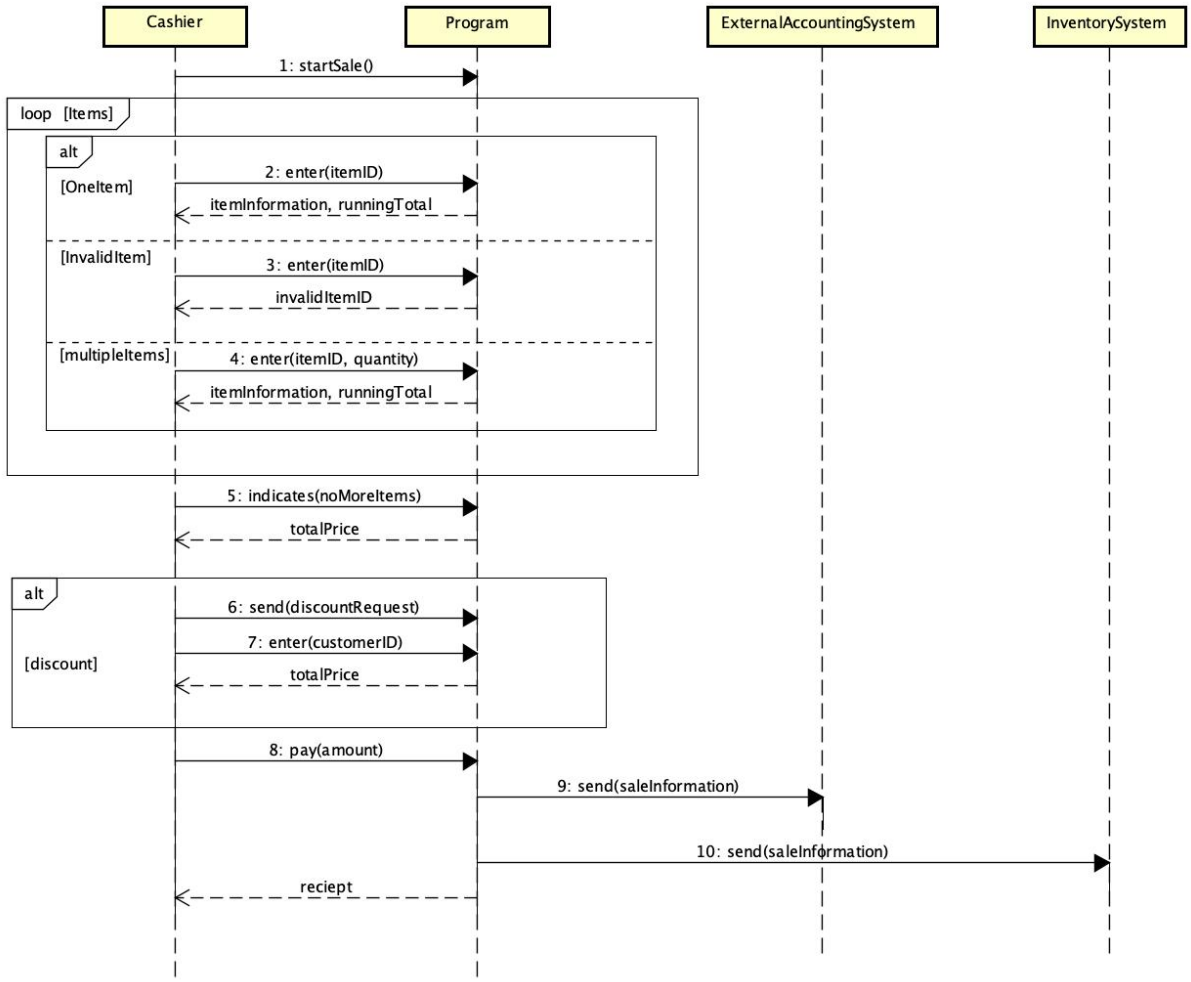
Slutliga versionen av domänmodellen till uppgift 1 visas nedan. Startpunkten är då ”customer hämtar product” och pilarna visar hur händelserna går till. Det mesta information som behövs under processen finns som attribut i klassen Computer, till exempel infromation angående produkten eller om kunden kan få rabatt. Däremot så hämtas en del information från klassen ItemInformation vilket innebär information bara om själva produkten till exempel pris och i fall en rabatt är inlagd för en specifik produkt.

Vi har ändrat domändiagrammet mycket under komplettringe i alla fall känner jag så kasske för att vi hade bättre förståelse och nu har vi även lagt till detaljer till exempel goods includes item men inget känns onödigt. Det finns fler än en klass som har fyra associationer vilket gör att spindel i när problemet inte finns.




Figur 3.1. Domänmodell (DM) som illustrerar beskrivningen ovan och scenariot i uppgift 1.

Slutliga versionen av sekventielldiagram till uppgift 2 visas nedan. Prickade linjer visar svarsmeddelandet till meddelandet som står ovanför den. Loop och if/else satserna visas i specifika rutor. När det gällde en händelse som skedde flera gånger användes. Vi har även inkluderat acounting och inventory system där de får information om sale för att kunna uppdateras.




Figur 3.2. sekventielldiagram(SSD) som illustrerar beskrivningen ovan och scenariot

# 4 Diskussion

Domändiagram i uppgift 1 är varken naiv eller programmatic DM eftersom ingen Program eller System som klass har använts, dessutom har det tagits hänsyn till andra information som inte finns i scenariot men man borde tänka på i verkligheten. Domändiagrammet har skapats på ett sätt att det är lätt att förstå om man följer pilarna och läsa vilket meddelande de har. Antal klasser har valts på ett sätt att det inte blir för många klasser som skapar förvirring men ändå tillräckligt många så att sambandet mellan de kan visas.

I SSD för uppgift 2 har det inte använts några objekt utan bara klasser. Detta var bland annat för att det skapades en klass Program vilket inte var tillåtet att ha i DM.

Svåraste med detta seminarium var att det fanns många olika tankesätt och idéer som kunde stämma vilket gjorde att man aldrig var säker om diagrammet är faktiskt rätt. Därför har vi genomfört det vi tyckte var rimligt och stämde med scenariot. Att diskutera med andra medans man löser uppgifter är också viktigt för att flera perspektiv och tankesätt kan komma fram.

Vi hade redan under peer review insett våra misstag när vi diskuterade och analyserade andras arbete så feedbacken vi fick var inte oväntat och vi förstod vad som skulle fixas. Som jag nämnde under metod, började vi med att diskutera vilka klasser som är onödiga och hur skulle vi minska antal associationer. ”Spindel i nät” problemet var alltid i våra tankar och det gjorde att vi ändrade diagrammen flera gånger för att till slut få något som verkade rimligt som inte hade ”spindel i nät” problem.

För SSD insåg vi att vi hade tänkt helt fel på föra seminarium och att det var egentligen mycket lättare än vad vi trodde. För båda seminarium fick vi hjälp mest för att få mer perspektiv till exempel ….