

WESTERN STYLE

Tobias Andersen, Casper Froberg Andersen, Stefan Krabbe Johansen, Mikkel Lindstrøm Paulsen, Arne George Ralston - *University College Nordjylland*, datamatiker 3. til 7. april 2017

Mock-ups

I dette afsnit vises den udviklede mock-up for systemet til Western Style. Mock-upen er vist på figur 1.

Western Style Ltd.

Opret Ordre

Se Ordre

Kunder

Lager

Opret Ordre

Ordrenummer: 0000001

slutdato: 12/6/17

Dato: 1/6/17

Status: Tilbud

Tilføj Vare

Varenummer: 0000123

Beskrivelse: Cowboystøvler lavet af ægte krokodilleskind

Antal: 1

Antal på lager: 79

Tilføj vare

Varenr	Gruppe	Beskrivelse	Antal	Pris stk	Pris deltotal	Rabat
Cowboy boots	Tøj	Krokodilleskind	1 par	10000	10000	10%

Redigere

Slet

Subtotal: 10000

Moms: 25%

Rabat: 10%

Samlet total: 11250

Gem Ordre

Kunde

Q Bias

Opret ny kunde

Navn:


Adresse:

Telefon:

By:

Email:

Gruppe:



Figur 1: Mock-up for opret ordre.

Tobias, Casper, Stefan, Mikkel, Arne

Side 1 af 8

Fully dressed use-case

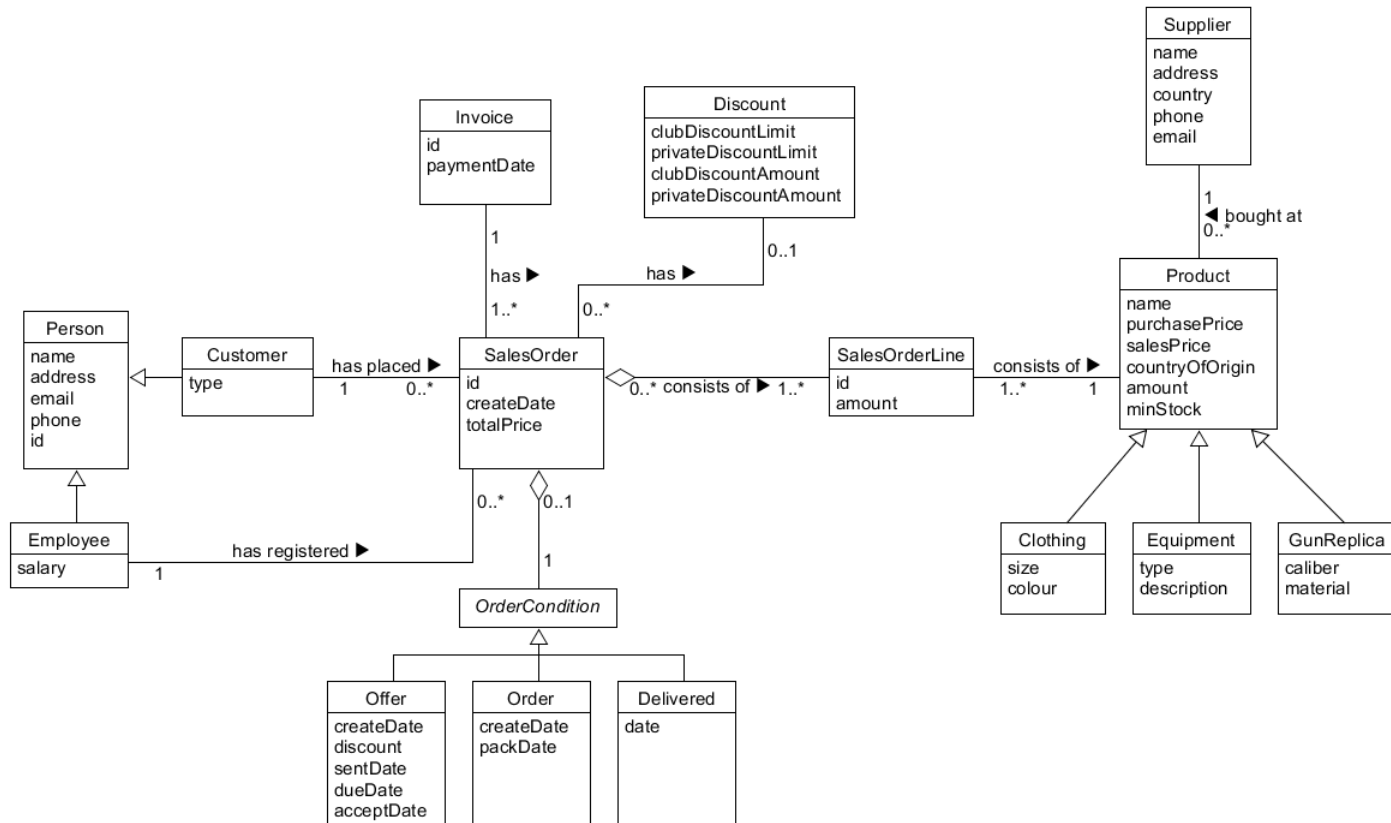
På tabel 1 vises en fully dressed beskrivelse af use-casen *Opret ordre*.

use-case	Opret ordre	
Aktører	Salgsmedarbejder	
Præbetingelse	Varen og kunden findes i systemet	
Postbetingelse	Den nye ordre er gemt i databasen med tilhørende attributter	
Frekvens	10 gange dagligt pr. medarbejder	
	Aktør	System
Flow of events	1. En kunde henvender sig for at bestille varer	
	2. Salgsmedarbejderen påbegynder en ny ordre	3. Systemet opretter en ny ordre
	4. Salgsmedarbejderen fremsøger kunden i systemet	5. Systemet viser kundens information
	6. Salgsmedarbejderen tilføjer kunden til ordren	7. Systemet associerer kunden til ordren
	8. Salgsmedarbejderen søger efter varen	9. Systemet viser varens navn og pris
	10. Salgsmedarbejderen tilføjer varen til ordren	11. Systemet associerer varen til ordren
	<i>Trin 8-11 gentages indtil alle de ønskede varer er tilføjet</i>	
	12. Salgsmedarbejderen afslutter ordren	13. Systemet tilføjer nødvendige attributter til ordren
		14. Systemet gemmer ordren
Alternative flow		5a. Kunden findes ikke i systemet 1. Systemet prompter at kunden skal oprettes
		9a. Varen findes ikke i systemet 1. Systemet prompter at varen skal oprettes
	10a. Salgsmedarbejderen har tilføjet forkert vare 1. Salgsmedarbejderen vælger hvilken vare der ønskes fjernet fra ordren	2. Systemet fjerner den valgte vare fra ordren
	12a. Salgsmedarbejderen annullerer ordren	1. Systemet nulstiller de tilføjede attributter og sletter den midlertidige ordre
		14a. Systemet har ikke forbindelse til databasen 1. Systemet viser en fejlmeddelelse

Tabel 1: Fully dressed for opret ordre

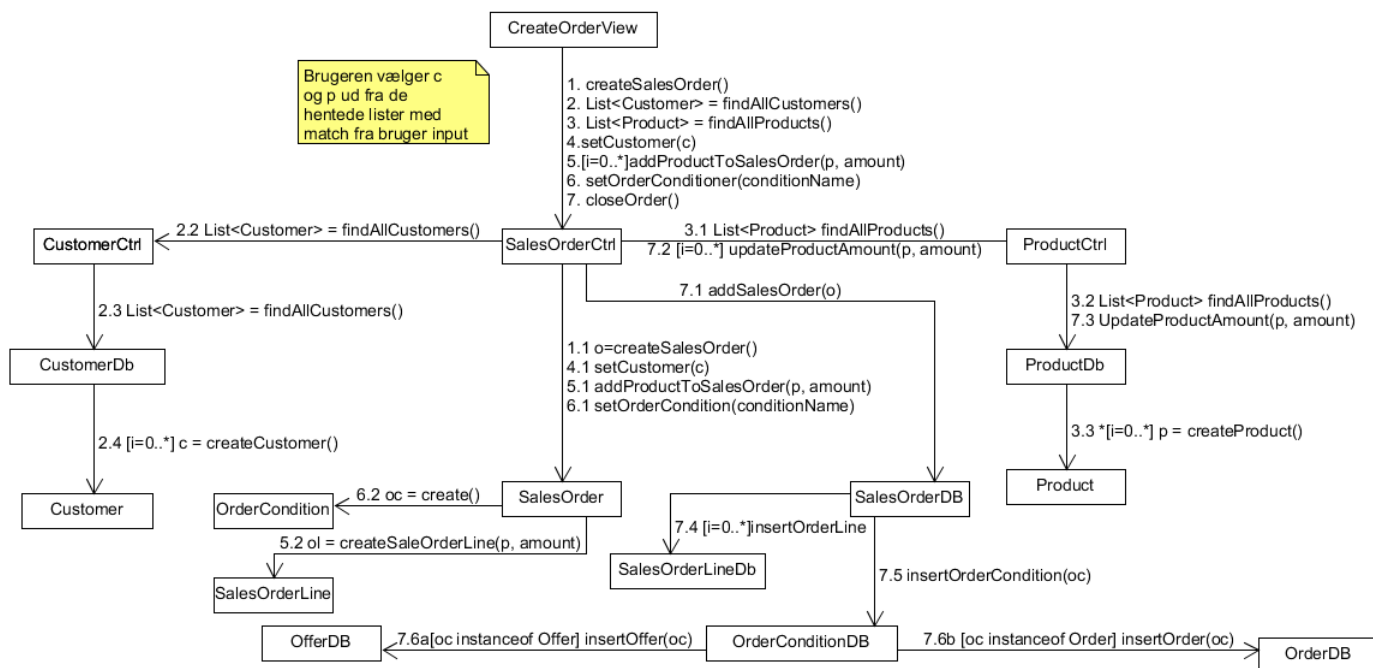
Domænemodel

I forhold til den originale domænemodel er klasserne Person, Employee, Discount, SalesOrderLine, OrderCondition, Offer, Order og Delivered blevet tilføjet. Dette er for at vise vi har lært noget i løbet af det første og andet semester. Desuden gør det at vi også har mulighed for at associere en ansat med en ordre samt en ordre kan være i flere forskellige tilstande. Derudover er det også muligt at tilføje rabat til en ordre og have flere forskellige ordrelinjer på en ordre.



Figur 2: Domænemodel for Western Style

Interaktionsdiagram



Figur 3: Interaktionsdiagram for Opret ordre

På figur 3 ses interaktionsdiagrammet for use casen *Opret ordre*. Først hentes lister op med Customer- og Product-objekter, hvilket muliggør at brugeren kan vælge kunde og vare fra en fremvist liste i brugerfladen. Den ønskede kunde vælges ud fra listen som derefter bliver associeret til SalesOrder-objektet via metodekaldet setCustomer(c). På samme måde udvælges de ønskede produkter som ved metodekaldet addProduktToOrder(p, amount) bliver associeret til et nyoprettet SalesOrderLine-objekter som dernæst associeres til det aktuelle SalesOrder-objekt. Brugeren vælger en order-status (OrderCondition) som enten kan være et tilbud eller ordre. Når ordren afsluttes gemmes SalesOrder, SalesOrderLine, og OrderCondition objekterne i databasen. For at gemme OrderCondition laves et type-tjek som vises i trin 7.6 som tjekker, hvilken sub-tabel objektet skal gemmes i. Som det ses er der implementeret dedikerede database-klasser til håndtering af indsættelse og hentning af alle SalesOrder referencer.

Test cases

Integrationstest

Acceptkriterier:

- Varen findes i systemet
- Ordren oprettes og gemmes i databasen
- Dato bliver korrekt registreret
- Rabat afhængig af købsbeløb bliver tilføjet korrekt (gratis levering)
- Ekstra rabat afhængig af købsbeløb rabat bliver regnet rigtigt

For at sikre, at programmet fungerer som det skal, har vi valgt at følgende metoder fra SalesOrder-klassen skal testes:

- checkDiscountPrivate()
- checkDiscountClub()

checkDiscountPrivate()

Regler:

- Hvis det samlede beløb er over 2500 DKK, kommer der rabat på leveringsomkostninger, som er 45 DKK.

Ækvivalente klasser:

- Discount
 - Klassen er valid så længe det samlede beløb for ordren er større end eller lig 2500 DKK
 - Klassen er ikke valid når ordrens samlede beløb er under 2500 DKK
- Forventet resultat for køb for 2500 DKK og over
 - Købet er valid
 - Købet får valideret klassen *Discount* og da det samlede beløb for ordren er over 2500 DKK, bliver der givet en rabat på 45 DKK
- Forventet resultat for køb under 2500 DKK
 - Købet er valid
 - Købet får ikke en *Discount*, da det samlede beløb for ordren er under 2500 DKK

Nr.	Beløbsgrænse for rabat	Ordre beløb	Forventet resultat (rabat i kr.)
1	2500	100	Afvist, rabat = 0 dkk
2	2500	1500	Afvist, rabat = 0 dkk
3	2500	2000	Afvist, rabat = 0 dkk
4	2500	2500	Ok, rabat = 45 dkk
5	2500	2800	Ok, rabat = 45 dkk
6	2500	3500	Ok, rabat = 45 dkk
7	2500	12000	Ok, rabat = 45 dkk

Figur 4: Test cases for checkPrivateDiscount

checkDiscountClub()

Regler:

- Hvis det samlede beløb er 1500 DKK eller derover, kommer der rabat på leveringsomkostninger, som er 45 DKK

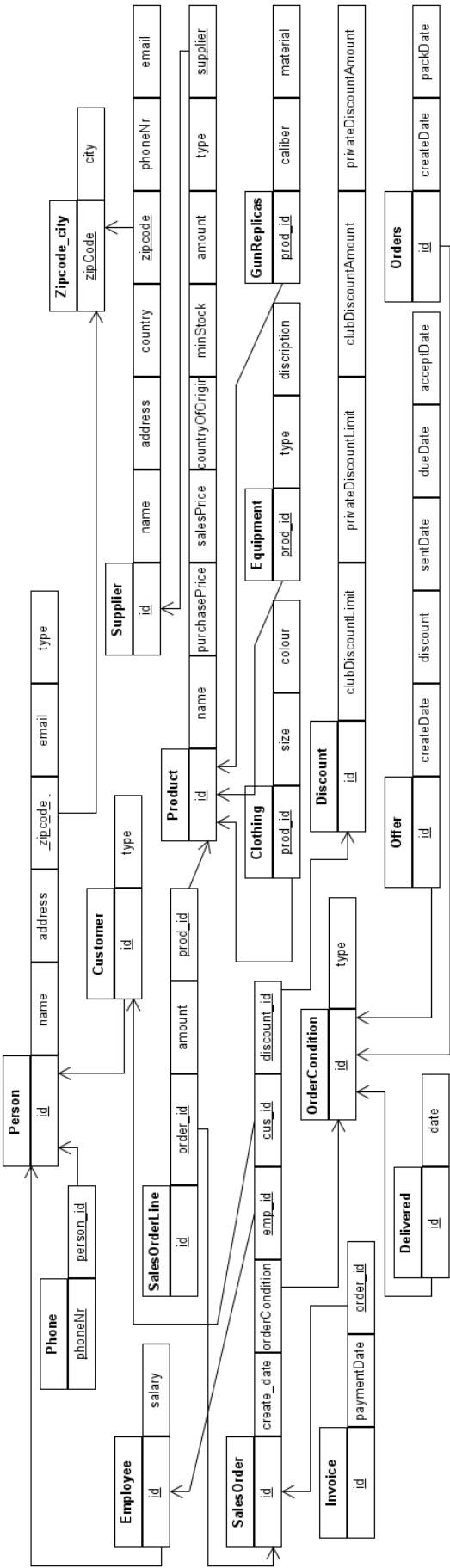
Ækvivalente klasser:

- Discount
 - Klassen er valid så længe det samlede ordrebeløb er 1500 DKK eller derover
 - Klassen er ikke valid når ordrens totale beløb er under 1500 DKK
- Forventet resultat for køb 1500 DKK og over
 - Købet er valid
 - Købet får en valid instans af klassen *Discount* og da ordrebeløbet er over 1500 DKK bliver der givet 45 DKK i rabat
- Forventet resultat for køb under 1500 DKK
 - Købet er valid
 - Købet får ikke en *Discount*, da ordrens totale beløb er under 1500 DKK

Nr	Beløbsgrænse for rabat	Ordre beløb	Forventet resultat (rabat i kr).
1	1500	100	Afvist, rabat = 0 dkk
2	1500	1500	Ok, rabat = 45 dkk
3	1500	2000	Ok, rabat = 45 dkk

Figur 5: Test cases for checkClubDiscount

Relationsmodel



Figur 6: Relationsmodel for Western Style

Modelleringen af databasen er gjort ud fra normaliseringsformerne op til og med Boyce-Codd normalform. Der er forsøgt at sikre funktionel afhængighed mellem attributterne og primærnøglen og derved mindske redundans. I person-tabellen ses dog et mindre brud på 1. normalform i og med at fornavn og efternavn er sammensat som én attribut. Dette er dog gjort for at simplificere og overskueliggøre med projekts begrænsede omfang i mente. Begge generaliseringer som ses i domænemodellen er i databasen designet således at forældre- og alle børne-klasserne får deres egen tabel (Generaliseringsmetode 1).

Kodestandard

Programmet følger Javas *Code Conventions*, som står beskrevet på Oracles hjemmeside. Dokumentet beskriver konventioner navne, fil organisering, indentering, kommentarer, erklæringer, statements, white space og programmerings praksis.