Kap3: Klassemodellering

HØGSKOLEN I BERGEN

I dag:

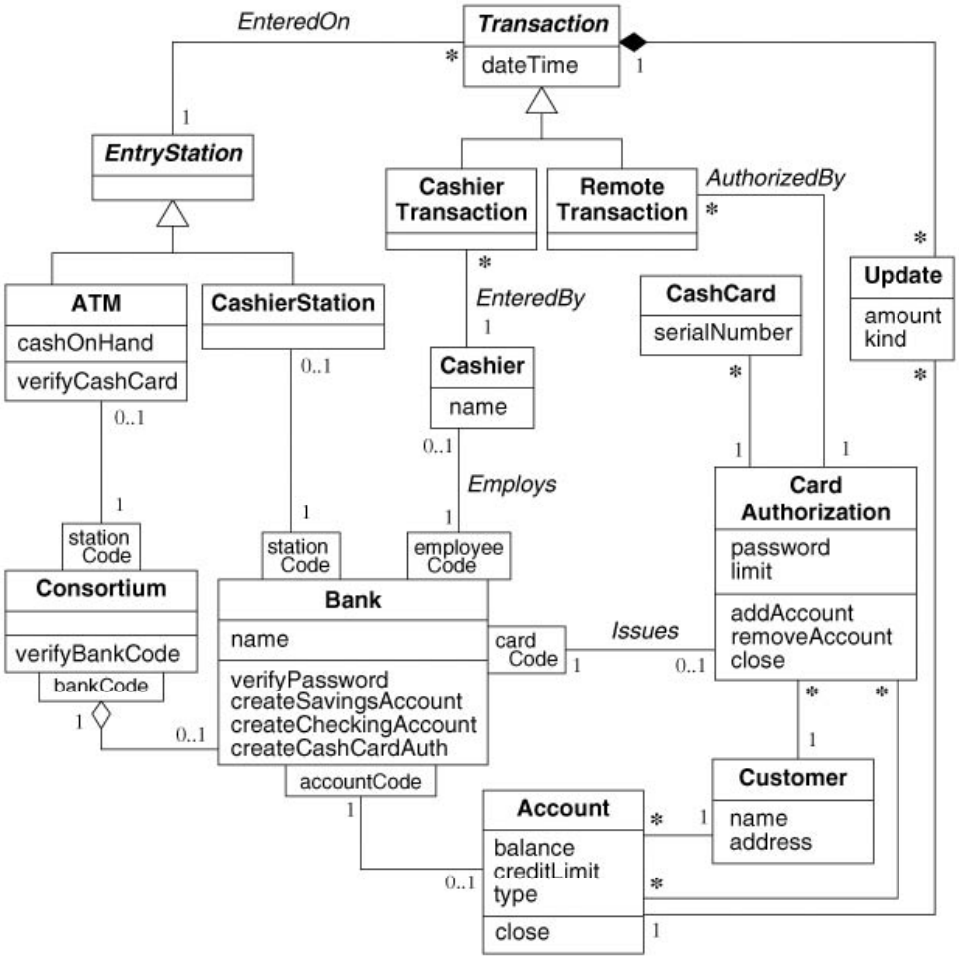
- Litt repetisjon fra sist (innledende om klassemodellen)
- Deretter… egentlig litt mer repetisjon, men nå fra intro-kurset i Java: Klasser og Objekt, Felt-/Instansvariabler og attributter. Metoder og Operasjoner…..
- UML notasjon for klasser, assosiasjoner osv….
- Generalisering/spesialisering og Arv
- OBS: Ikke pensum: 3.5.1 – 3.5.3

Fra sist…

HØGSKOLEN I BERGEN

Klassemodellen

- Beskriver den statiske struktur av objekter, deres relasjoner, attributter og operasjoner:



– Klasser representerer sentrale begreper fra domenet.
– Generalisering brukes for å dele struktur og oppførsel mellom klasser.
– Utgjør kontekst og grunnlag for diagram innen de andre modellene
– Klassemodellen er den viktigste av de tre modellene

..og den gir en intuitiv grafisk representasjon som også er verdifull for kommunikasjon mot brukere/oppdragsgivere/kunder

Klassediagram

- Gir en grafisk notasjon for modellering av klasser og sammenhengen mellom disse
- Nyttige for både abstrakt modellering og for å designe faktisk program

Objektdiagram

- Viser eksempel på klasse + individuelle objekt og deres eventuelle relasjoner

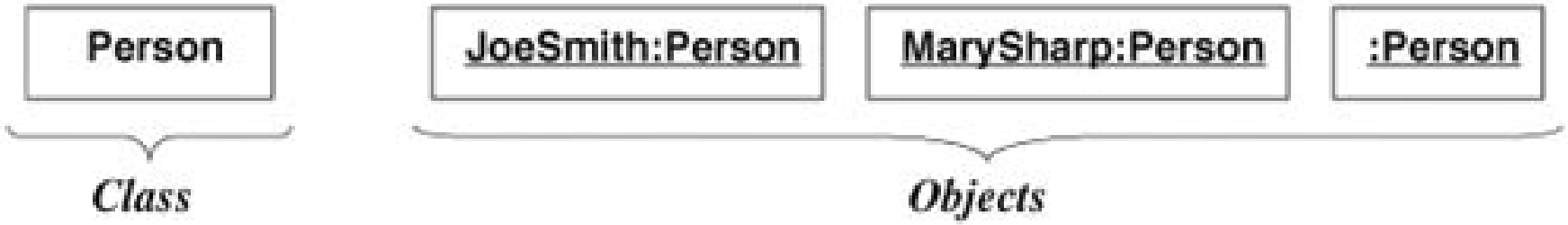


Figure 3.1 A class and objects. Objects and classes are the focus of class modeling.

Objekt og Klasse

Objekt:

- Er et konsept, en abstraksjon eller en ting med identitet som har en betydning for en applikasjon
- Alle objekt har identiteter og kan skilles fra hverandre. Eks: to helt identiske epler vil fortsatt være to individuelle epler.
- Og – et objekt er en instans av en **Klasse**

Klasse:

- Beskriver en gruppe av objekt med de samme egenskaper (attributter/instansvariabler), den samme adferd (operasjoner), de samme relasjoner til objekt av både egen og andre klasser, og den samme semantikk.
- Vi abstraherer problem ved å gruppere objekt i klasser.

Klassediagram & Objektdiagram

HØGSKOLEN I BERGEN

Attributter og verdier

- Merk at *Verdi* står til *Attributt* som *Objekt* står til *Klasse*
- Merk dere også at UMLs notasjonsform for Klasse har en hvis likhet med hvordan vi definerer en klasse i Java (som er et objektorientert språk…….)

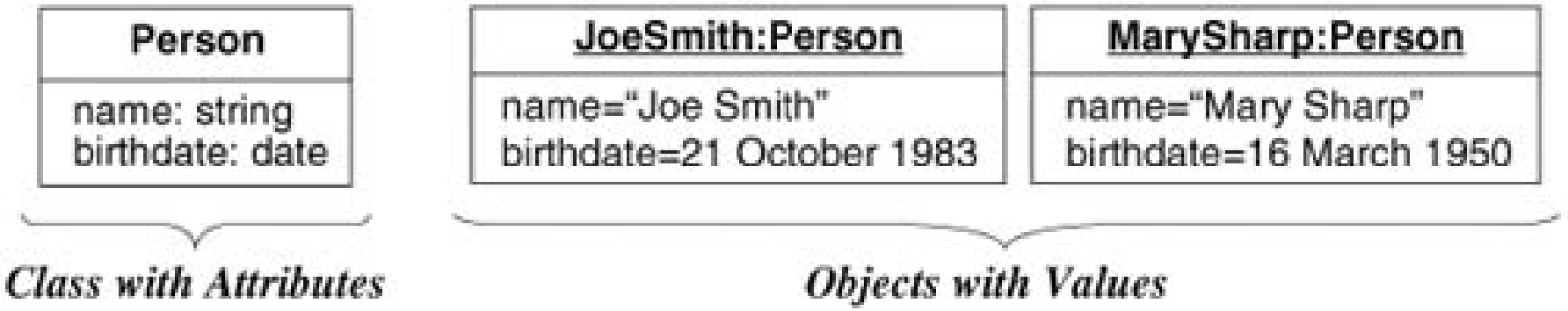


Figure 3.2 Attributes and values. Attributes elaborate classes.

Operasjoner og Metoder

HØGSKOLEN I BERGEN

Operasjoner

- De funksjoner (prosedyrer) som en legger til en klasse. Det er disse som gir adferd til objekt av klassen.
- Alle objekt av en klasse deler de samme operasjonene.
- Kan også ha at den samme operasjonen deles av flere klasser. Operasjonen er da polymorfisk.

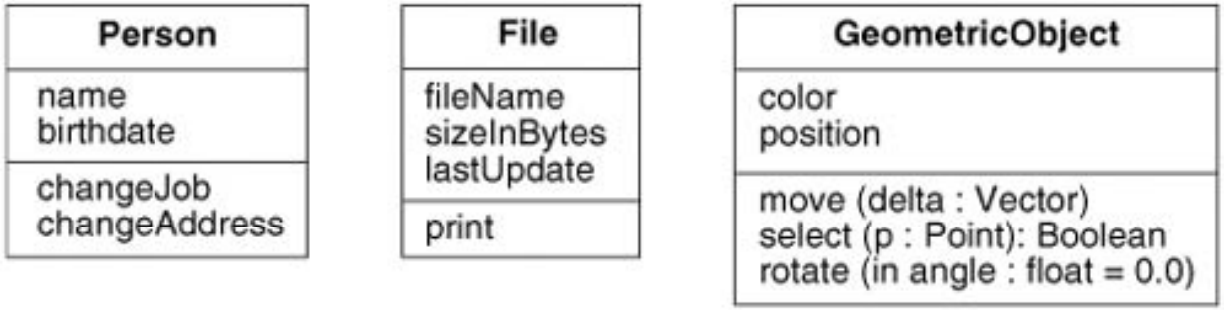


Figure 3.4 Operations. An operation is a function or procedure that may be applied to or by objects in a class.

Metoder

- Den faktisk implementeringen av en operasjon – for eksempel i Java.

Notasjon for klasse

HØGSKOLEN I BERGEN

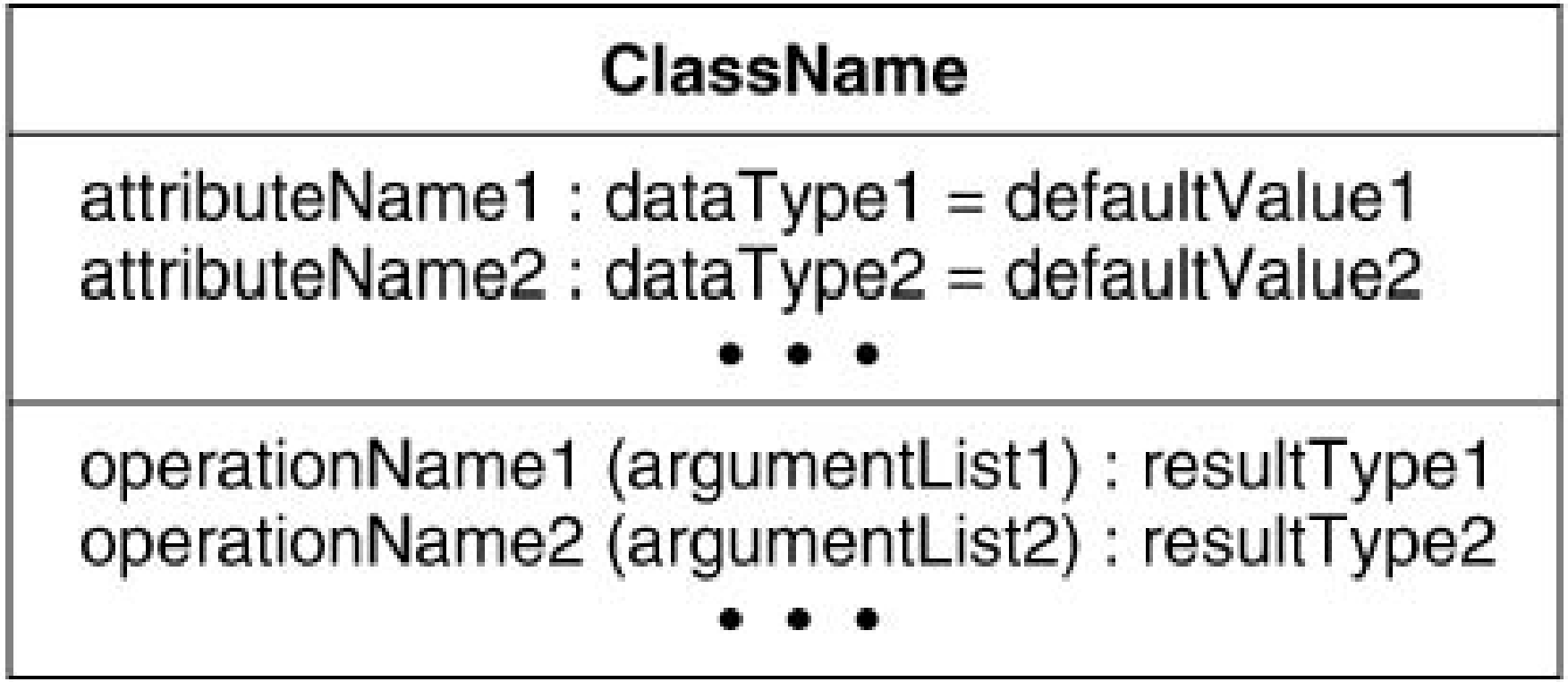


Figure 3.5 Summary of modeling notation for classes. A box represents a class and may have as many as three compartments.

Linker og Assosiasjoner

HØGSKOLEN I BERGEN

Link

- Representerer den fysiske (for eksempel Bok og Kapittel) eller konseptuelle (for eksempel Company og Person) forbindelsen mellom objekt.

Assosiasjon

- Representerer en beskrivelse av en gruppe linker med felles struktur og felles semantikk.
- Husk at en assosiasjon kan leses begge veier.

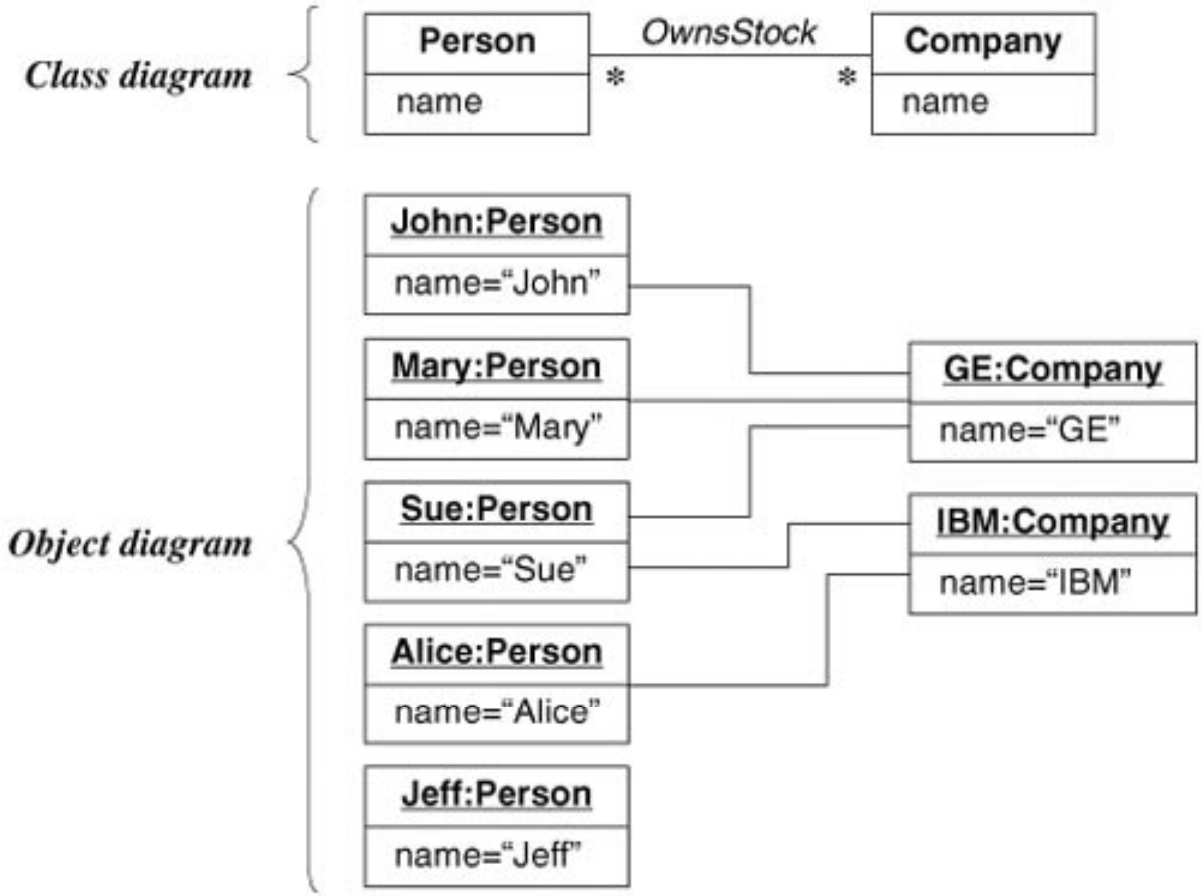


Figure 3.7 Many-to-many association. An association describes a set of potential links in the same way that a class describes a set of potential objects.

Multiplisitet

- Spesifiserer det antall av instanser fra en klasse som kan relateres til en enkelt instans i en assosiert klasse.
- Mulige notasjoner:
  - 1
  - m
  - *
  - 1..1
  - 0..m
  - 1..m
  - 1..*
  - 2..5
  - osv…

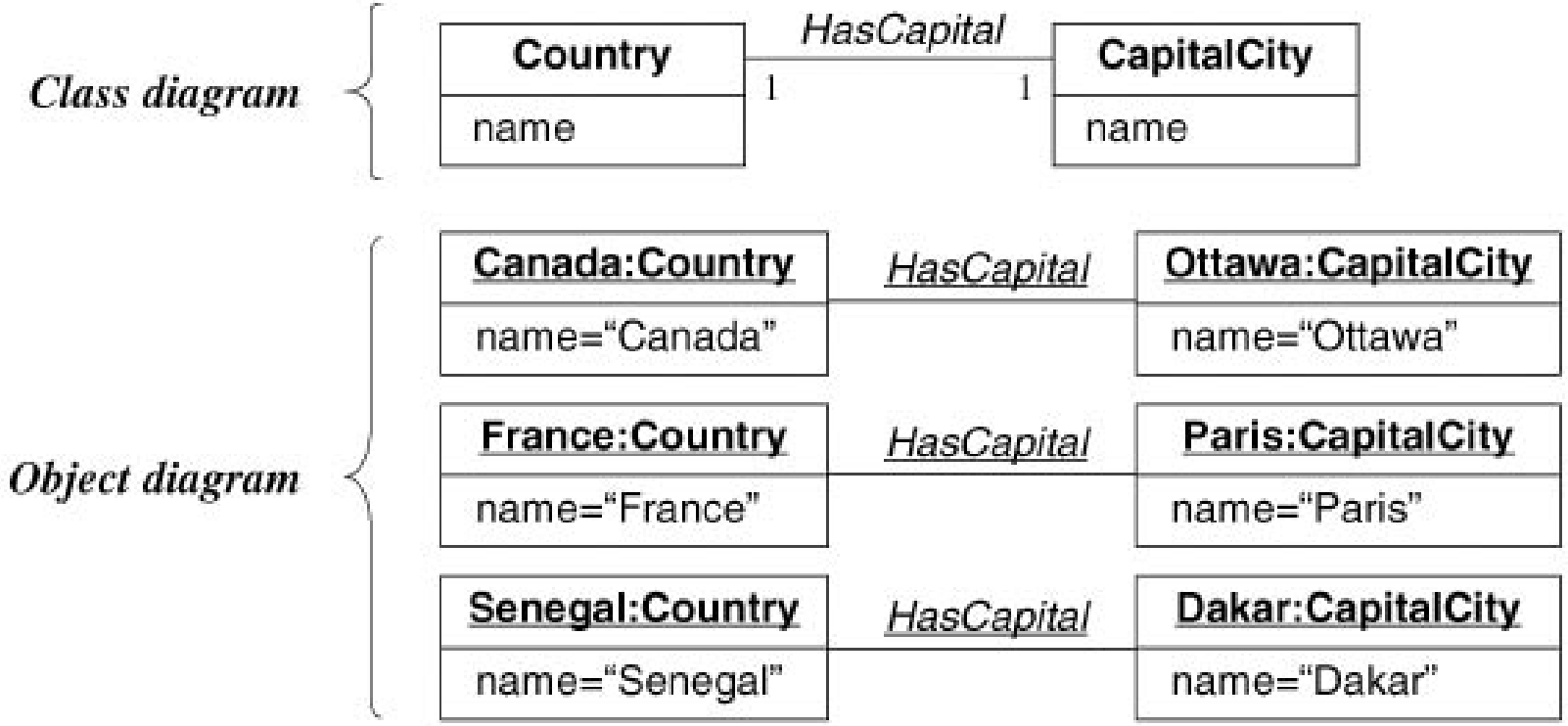


Figure 3.8 One-to-one association. Multiplicity specifies the number of instances of one class that may relate to a single instance of an associated class.

Assosiasjoner og notasjoner, flere eks.

HØGSKOLEN I BERGEN

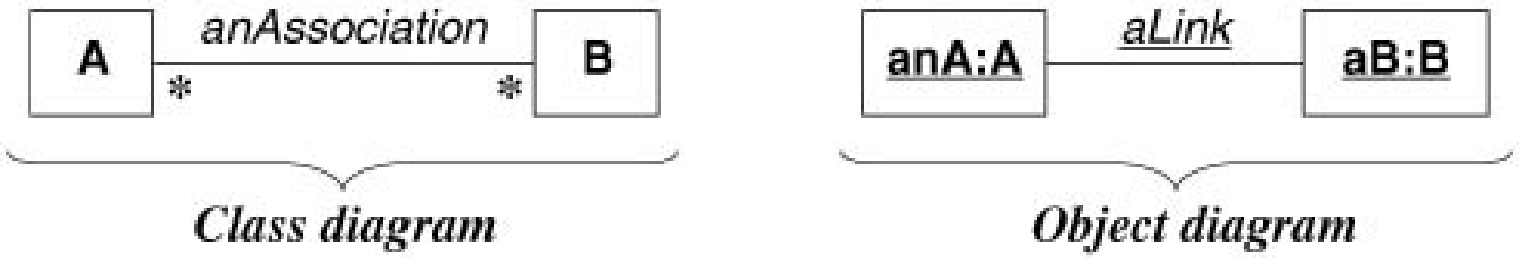


Figure 3.10 Association vs. link. A pair of objects can be instantiated at most once per association (except for bags and sequences).

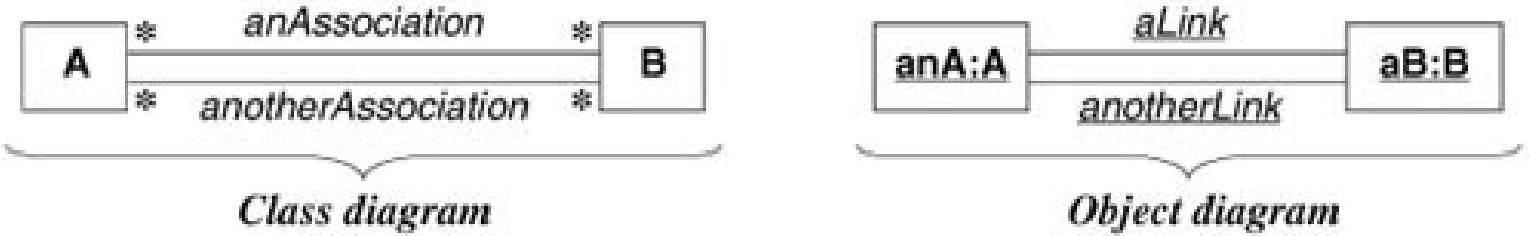


Figure 3.11 Association vs. link. You can use multiple associations to model multiple links between the same objects.

Assosiasjoners sluttnavn

- Hjelper med å sette roller til klasser i en assosiasjon
- Er en absolutt nødvendighet for assosiasjoner mellom to objekt av samme klasse

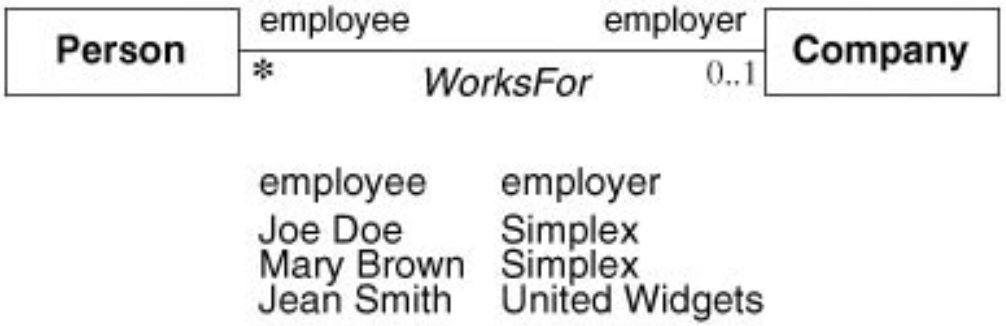


Figure 3.12 Association end names. Each end of an association can have a name.

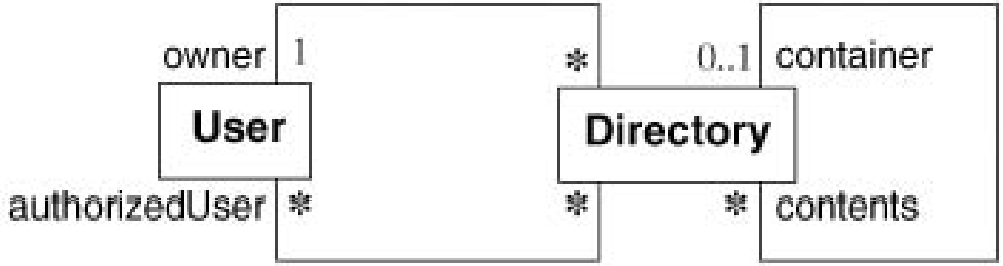


Figure 3.13 Association end names. Association end names are necessary for associations between two objects of the same class. They can also distinguish multiple associations between a pair of classes.

Assosiasjoners sluttnavn, et eks. til..

HØGSKOLEN I BERGEN

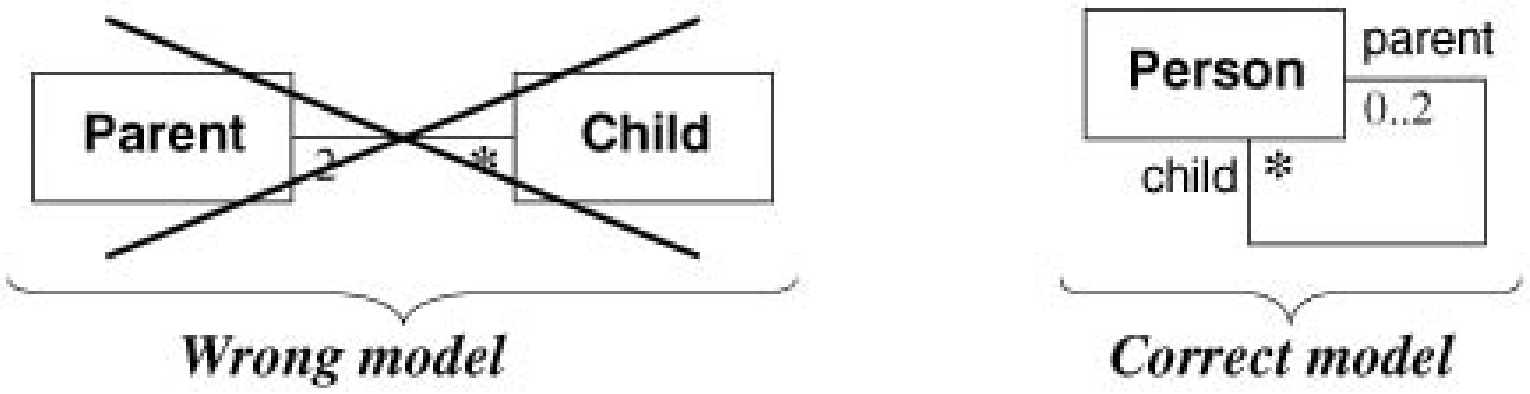


Figure 3.14 Association end names. Use association end names to model multiple references to the same class.

Orden og sekvens i mange assosiasjoner

Orden

- Angir en form for prioritering, for eksempel at bare øverste vindu er synlig på en skjerm (se under)

Figure 3.15 Ordering the objects for an association end. Ordering sometimes occurs for "many" multiplicity.

Sekvens

- Angir en ordnet samling hvor også duplikater er lovlig (se under)

Figure 3.16 An example of a sequence. An itinerary may visit multiple airports, so you should use {sequence} and not {ordered}.

Assosiasjonsklasse

- En assosiasjon som også er en klasse
- Har attributter som hører til linken og ikke til noen av klassene

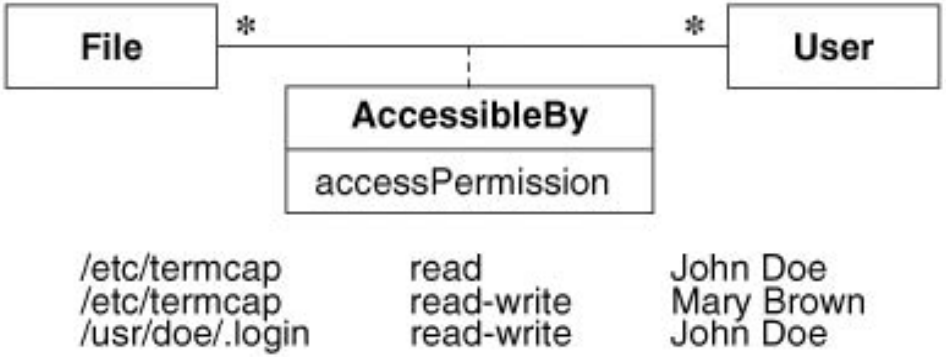


Figure 3.17 An association class. The links of an association can have attributes.

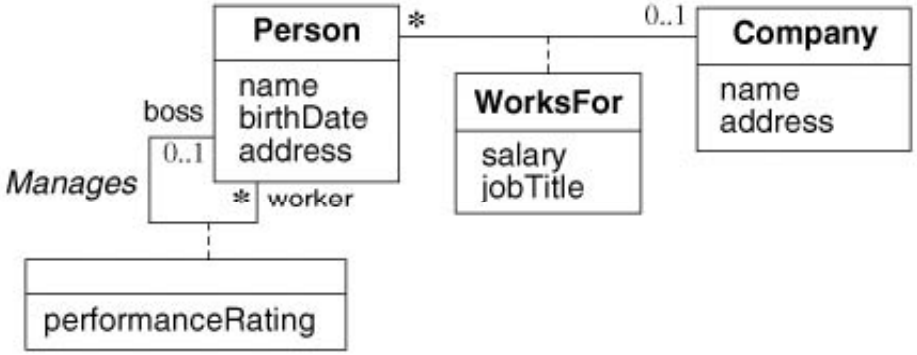


Figure 3.18 Association classes. Attributes may also occur for one-to-many and one-to-one associations.

Assosiasjonsklasse, litt mer…

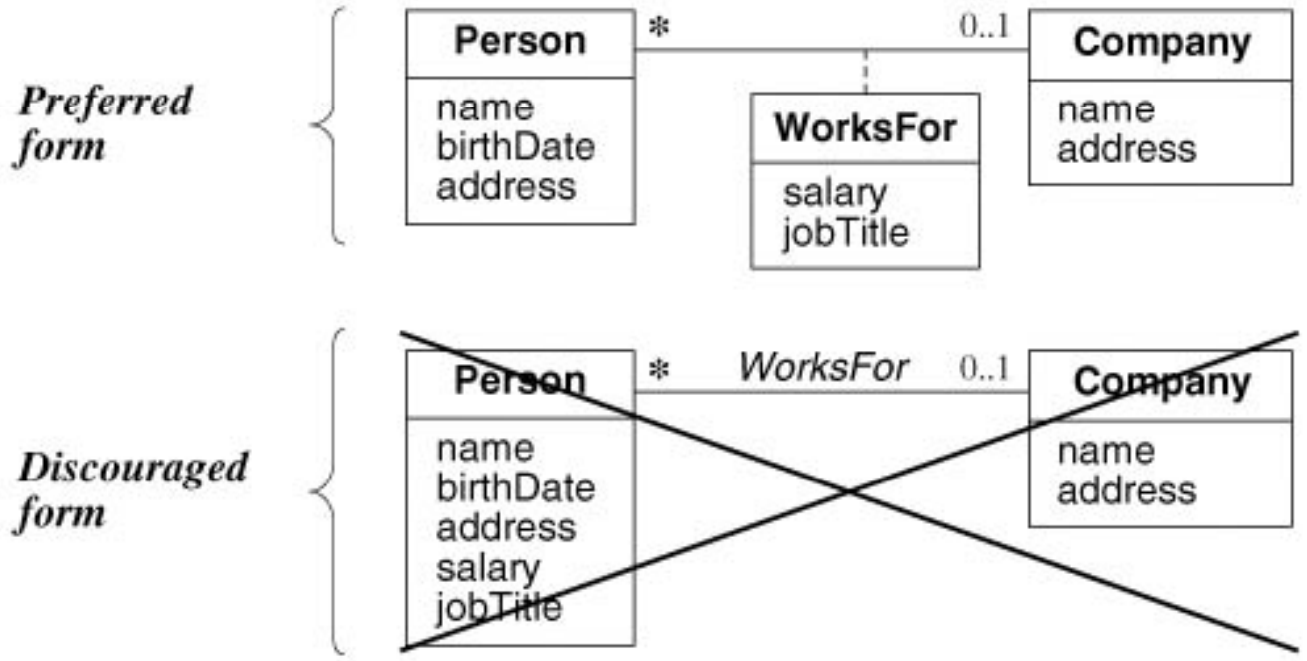


Figure 3.19 Proper use of association classes. Do not fold attributes of an association into a class.

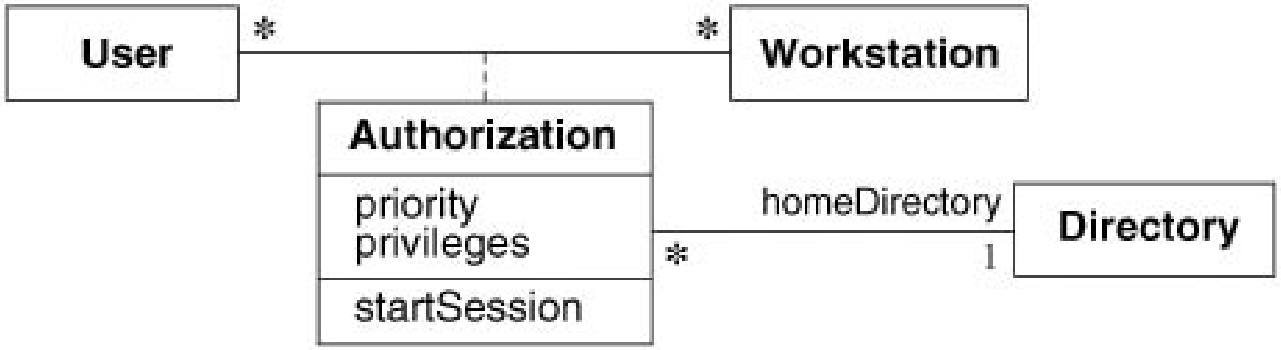


Figure 3.20 An association class participating in an association. Association classes let you specify identity and navigation paths precisely.

Kvalifisert assosiasjon

- En assosiasjon hvor et attributt kalt qualifier kvalifiserer et objekt for medlemskap i assosiasjonen

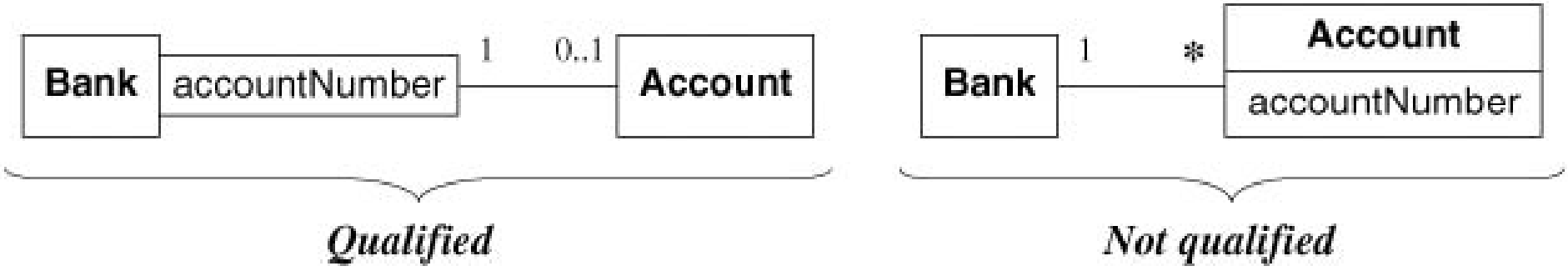


Figure 3.22 Qualified association. Qualification increases the precision of a model.

Generalisering/spesialisering og arv

HØGSKOLEN I BERGEN

- **Generalisering**
  - Når en ser at to eller flere klasser har flere ting til felles hva gjelder både attributter, operasjoner på disse og assosiasjoner mot andre klasser.
  - Samler da det som er felles i en felles superklasse.
  - De enkelte klassenes særegenheter hva gjelder attributter, operasjoner og assosiasjoner legges da til de opprinnelige klassene som kalles subklasser.
  - Subklassene **arver** så alt av attributter, operasjoner og assosiasjoner fra superklassen.

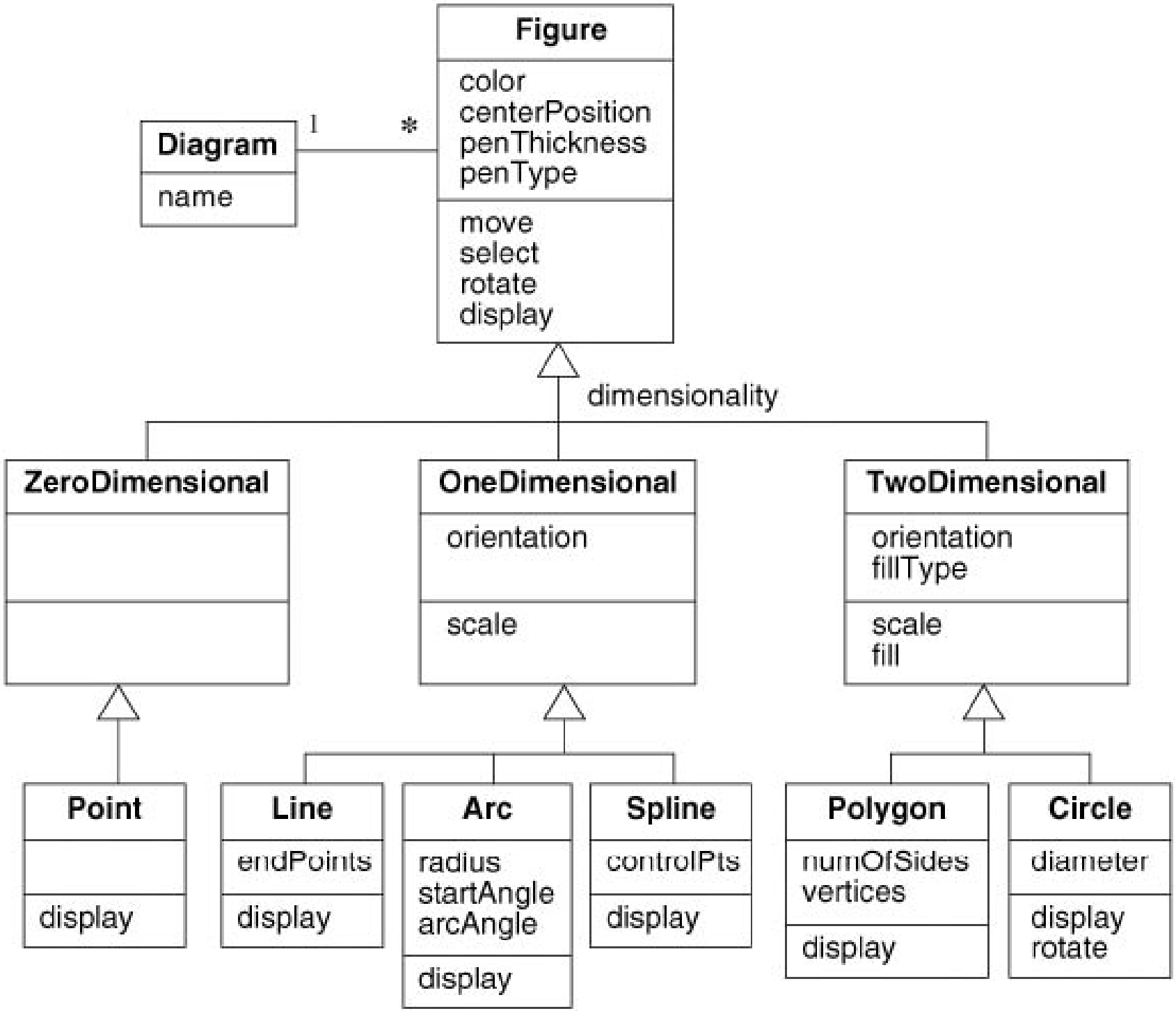


Figure 3.25 Inheritance for graphic figures. Each subclass inherits the attributes, operations, and associations of its superclasses.

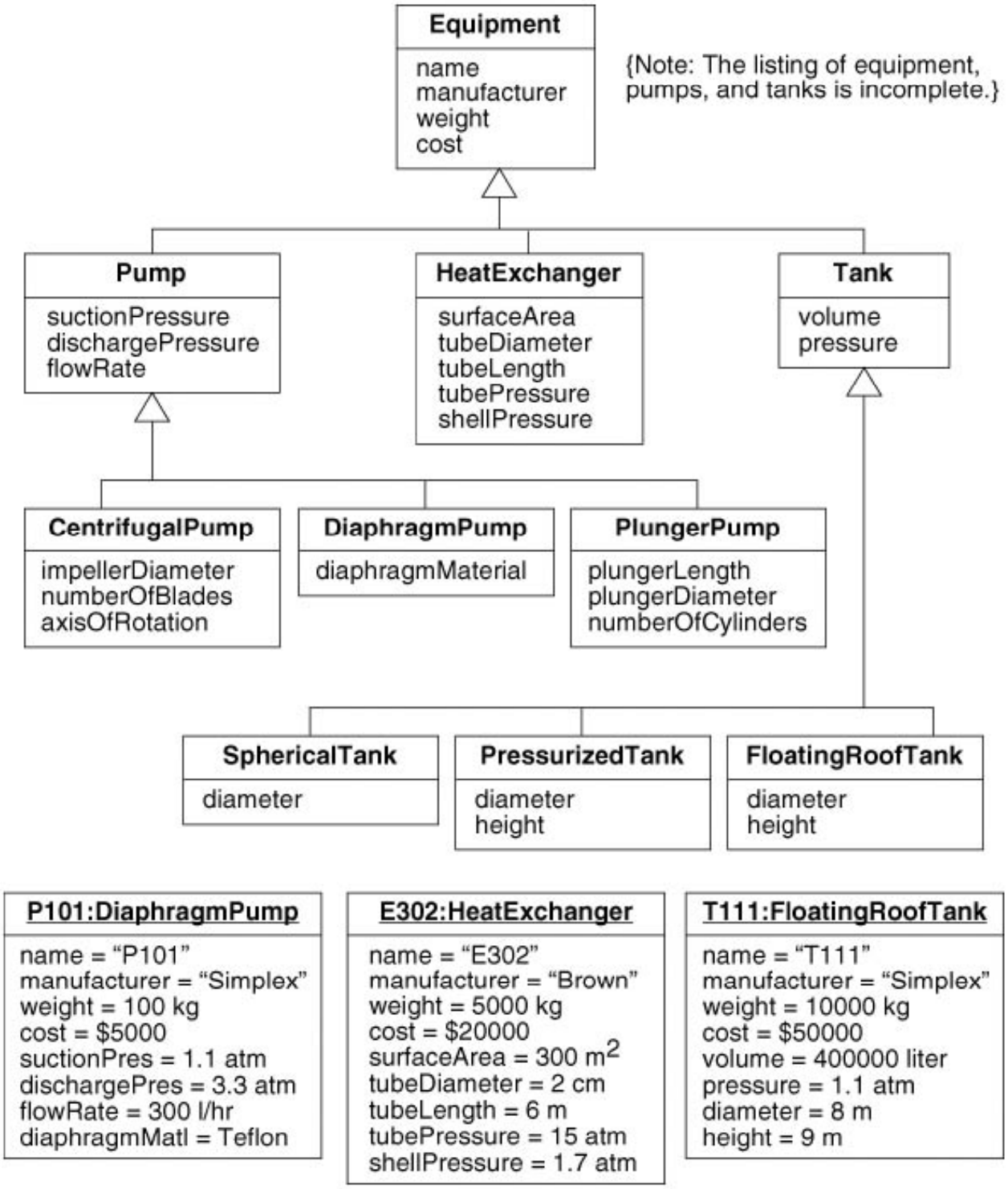


Figure 3.24 A multilevel inheritance hierarchy with instances. Generalization organizes classes by their similarities and differences, structuring the description of objects.

Generalisering/spesialisering og arv, forts.

HØGSKOLEN I BERGEN

- **Spesialisering**
  - Motsatt vei av generalisering. En har en opprinnelig klasse, men føler behov for å avlede denne til en eller flere spesialiseringer.
  - Beholder det som er felles i superklassen
  - Lar så subklasser av superklassen inneholde attributter, operasjoner og assosiasjoner som er nødvendige for de ønskede spesialiseringer.
  - Tenk: Har oppfunnet bilen, men finner så ut at det kan være behov for både personbiler, lastebiler og busser. Mye er felles for de tre nye sublassene og legges i en superklasse BIL, mens de tre subklassene så får innhold for spesialiseringer.

Generaliserings/spes. hierarkier

HØGSKOLEN I BERGEN

- Enkle generaliseringer/spesialiseringer organiserer klasser gjennom et hierarki:
  - Hver subklasse har en og bare en superklasse
  - En superklasse vil naturlig nok ha (to eller) flere subklasser direkte under seg
  - En subklasse kan så være superklasse for nye subklasser (flere nivå av generalisering/spesialisering)
  - En subklasse (etterfølger) arver fra sine foreldre/superklasser (forfedre) på alle nivå over
  - Generaliserings/spesialiserings-sett navn: Kan sette beskrivende attributtnavn på generalisering/spesialisering
  - Notasjon nærmest som et tre
  - **Eksempel**: se fig. 3.24 og 3.25

Generaliserings/spes. hierarkier, forts.

HØGSKOLEN I BERGEN

- **Hvorfor Generalisering/Spesialisering:**
  - Støtter polymorfisme: Inn med ny subklasse og mye av ”dennes kode” er allerede tilgjengelig gjennom arv.
  - Bedre struktur og oversiktlighet i kode: oppretter en taxonomi samt organisering av objekter på bakgrunn av likheter og forskjeller
  - Gjenbruk av kode: Hvorfor skrive på nytt…..?
- **Dybde på hierarkier?**
  - For dype gir gjerne mindre lesbarhet
    men, av og til behov for en hvis dybde
  - Tommelfingerregel: unngå dypere enn 5-6 nivåer

Arv og overskriving (/utviding)

HØGSKOLEN I BERGEN

- Overskriving (override)
  - En subklasse kan overskrive en egenskap fra superklasse ved å definere denne på nytt (bruke samme navn) i subklassen.
  - Subklassens definisjon gjennom overskriving vil alltid gjelde over den i utgangspunktet arvede (likt navn) fra superklassen.
- Utvidelse
  - En subklasse kan også bygge videre på en superklasses egenskap gjennom en utvidelse av denne.
  - Lært på første javakurset?
- OBS:
  - Ikke foreta endringer av grensesnitt: eks. metodekall

De to siste UML eksemplene fra kap. 4

HØGSKOLEN I BERGEN

Fig 3.26: Klassemodell for et vindussystem

- Forskjellige typer vindu for et vindussystem på en datamaskin, ink. komponenter for de enkelte vindu
- Basert på arv
- Merk klasse ScrollingCanvas som er basert på dobbel arv (både fra klasse ScrollingWindow og klasse Canvas)
- Forsøk å forstå (lese) diagrammet

Fig 3.27: Klassemodell for kredittkort og kontoer

- Basert på arv
- En institusjon kan utstede mange kredittkort, hvert enkelt identifisert ved et kontonummer (kvalifiserer)
- Kan ha flere (samme adresse/ i familie) med samme kredittkort
- Har så (måneds?)-utskrifter for hvert kort inneholdende forskjellige transaksjoner

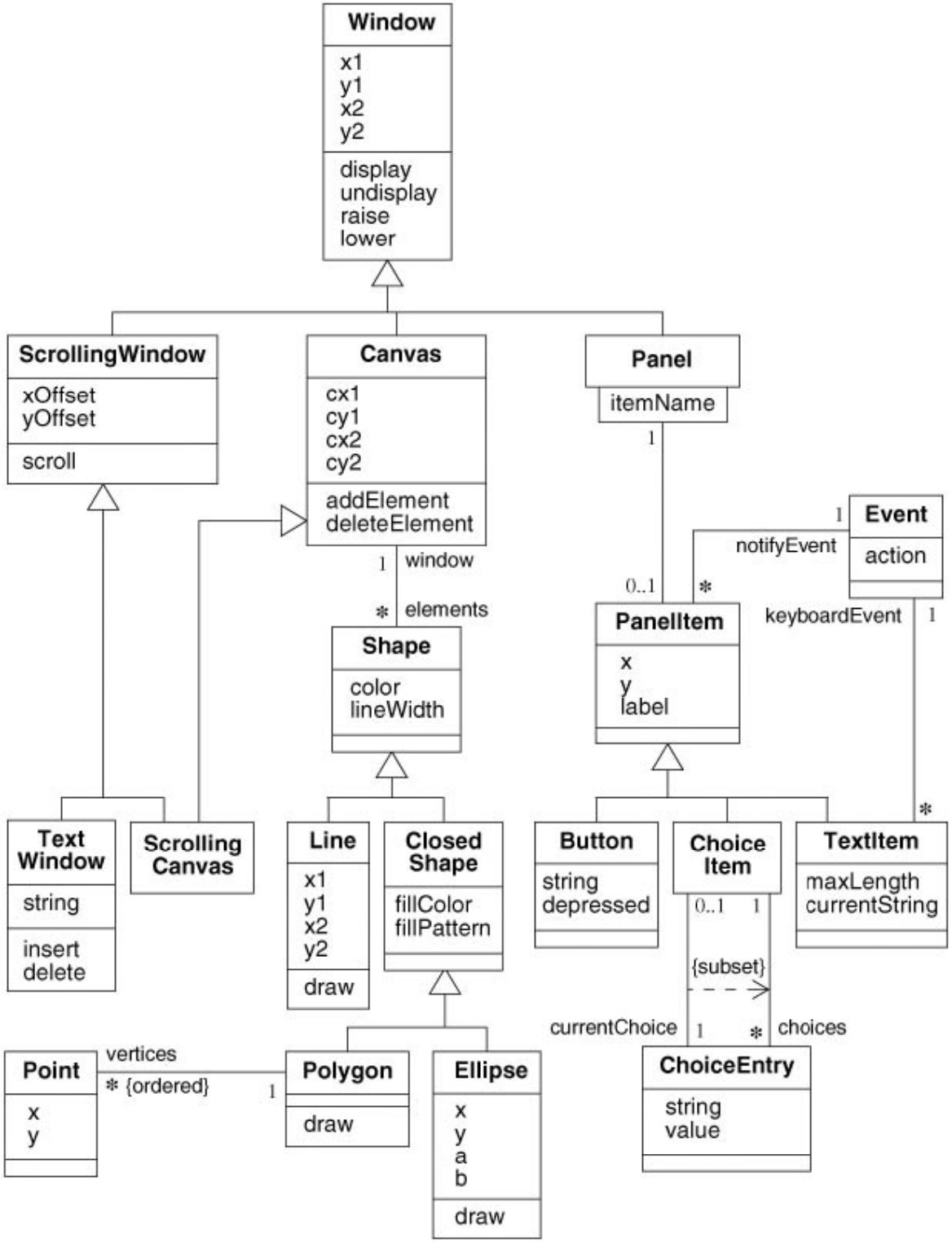


Figure 3.26 Class model of a windowing system.

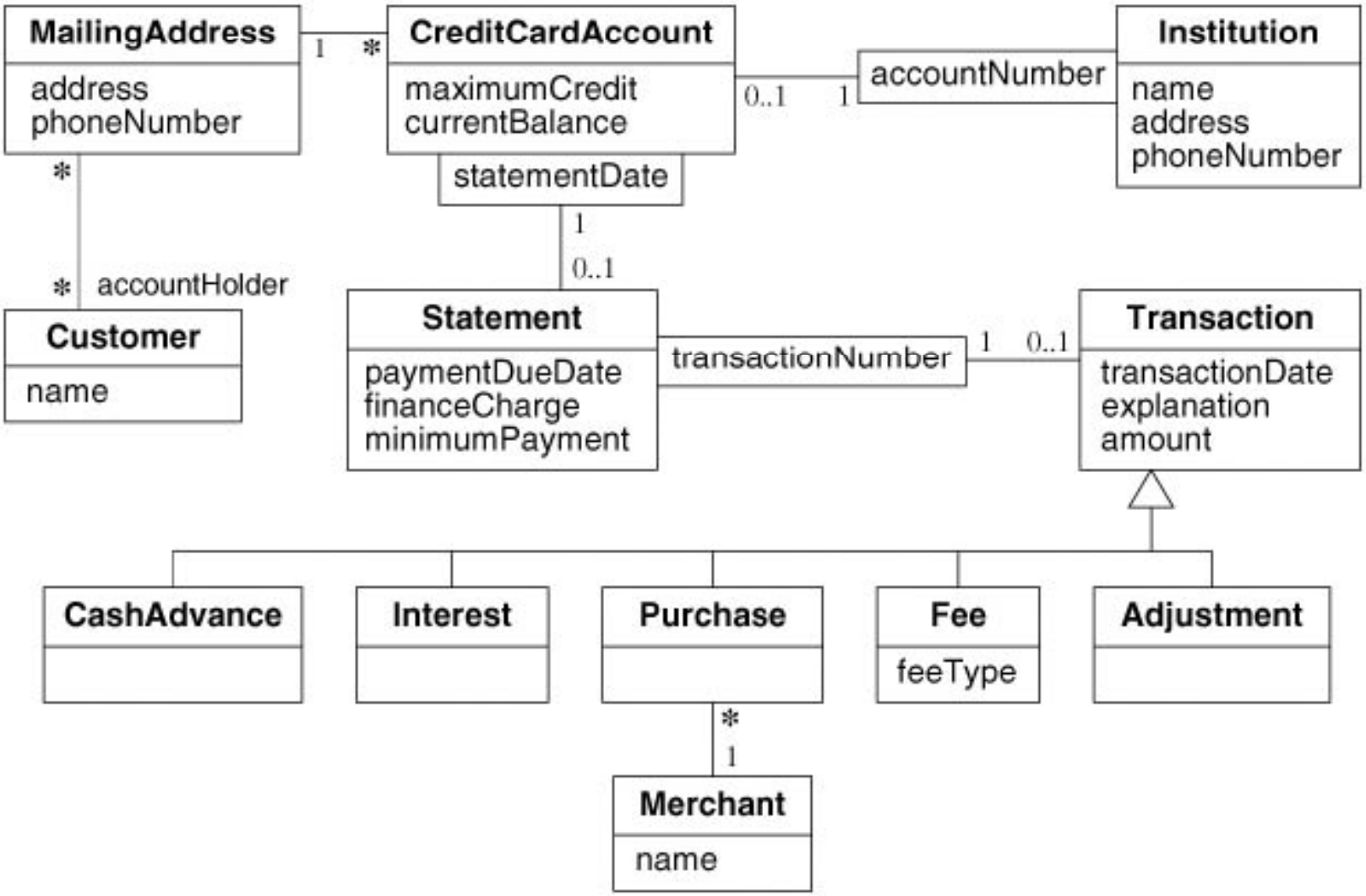


Figure 3.27 Class model for managing credit card accounts.

Før vi forlater kap. 3

HØGSKOLEN I BERGEN

Husk å jobbe med stoffet:

- Oppgave 3.1
- Oppgave 3.2
- Oppgave 3.13a

  Klassediagram for følgende klasser med assosiasjoner:

  – skole, lekeplass, rektor, skoleråd, klasserom, bok, student, lærer, kafeteria, hvilerom, røykerom, PC, pult, stol, dør, disse
- Oppgave 3.15
- Det anbefales at dere gjør flest mulig av oppgavene i boken (hver for dere, diskuter deretter løsninger med andre).