Programutvecklingsmetodik

Stegen vid programutveckling är typiskt

- kravspecifikation
- analys
  - ska ge en modell av systemet som är mer exakt och en specifikation som är mer fullständig
  - ge en bättre förståelse av systemet och dess relation till omvärlden
- design
  - förfining av analysen
  - utgöra underlag för kodning
- kodning
- vidareutveckling och underhåll

Objektorienterad metodik

Objektorienterade metoder är typiskt

- byggda kring användningsfall – ”use cases”
- inkrementella och iterativa – moment upprepas och förfinas

Agila metoder (”lättrörliga”) har vunnit stor popularitet under senare år

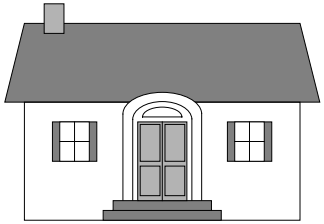
- utveckling sker i nära samarbete med kunden
- utvecklingen bedrivs inkrementellt och iterativt och med regelbundna små delleveranser
- saker utvärderas löpande och kan ändras för att möta nya krav och önskemål

Objektmodellen

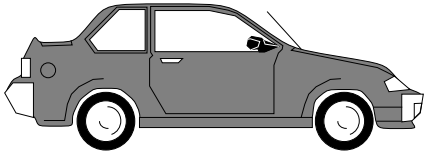
- objekt bildar den strukturella grunden
  - program består av objekt som kommunicerar med varandra
- underlättar återanvändning
  - kan försvåra underhåll
- en naturlig begreppsmodell används
  - abstraktion
  - inkapsling
  - klassificering
  - hierarkisk strukturering
  - generalisering/specialisering

Varje objekt har en unik identitet

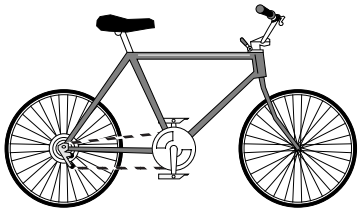
- varje objekt måste kunna särskiljas från alla andra objekt



Pers hus



Annas bil

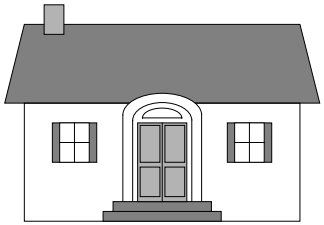


Eriks cykel

Stinas dator

Objekt har attribut

- attribut avser gemensamma egenskaper för en viss typ av objekt
- ett objekts egenskaper beskrivs av dess attributs värden

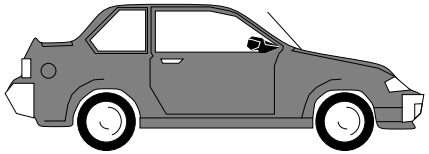


hustyp: villa

antal_rum: 5

taxeringsvärde: 765.000

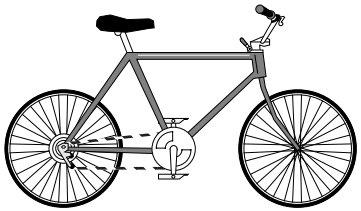
ägare: Per



fabrikat: Volvo

regnr: ABC123

ägare: Anna



fabrikat: Crescent

modell: terrängcykel

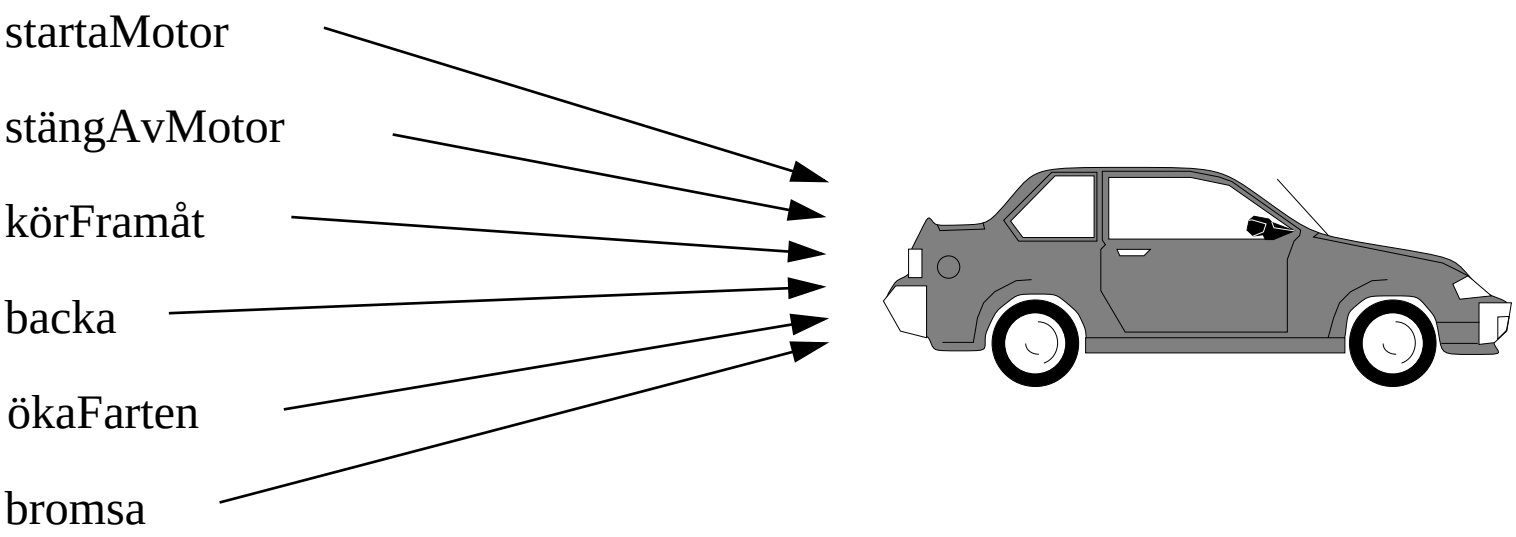
ramnummer: MBC1024718

fabrikat: Macintosh

serienummer: MAC239008761

Objekt har beteende

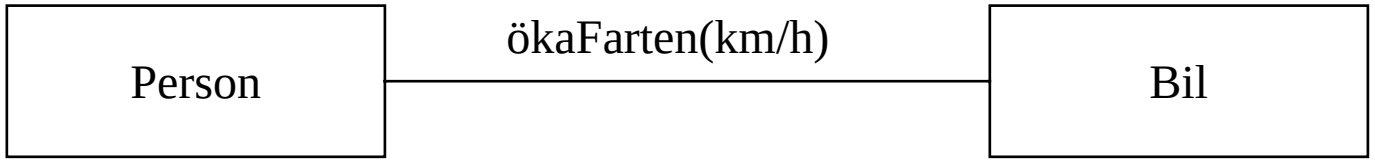
- beteende avser de tjänster som ett objekt ställer till förfogande för andra



Objekt samarbetar

Objekt sägs kommunicera genom *meddelanden*

- definierar interaktion mellan objekt
- identifierar beroenden (associationer) mellan objekt
- står för informationsöverföring



Hur hittar man objekt?

- sök i problemområdet
  - analysera kravspecifikationen
  - ”brainstorming”
  - checklista
  - …
- substantiv i kravspecifikationen eller i beskrivningar av användningsfall
- objekt kan finnas i många former
  - faktiska ting
  - roller
  - händelser
  - interaktioner
  - begrepp
  - information som måste lagras
  - …

Definiera ansvar för objekt

- bra objekt/klasser gör *en* sak bra
  - kan vara en komplex sak
  - kan krävas flera deloperationer för att utföra
  - kan kräva samarbete med andra objekt
- objekt ska ha dataattribut och operationer
  - motiverar objekts existens
- hur hittar man ansvar?
  - adjektiv i kravspecifikationen
  - vid analys av användningsfall
  - beskrivningen av ett objektet
- hur hittar man operationer, tjänster?
  - verb och verbfraser i kravspecifikationen
  - vid analys av användningsfall

Beskriv klasserna

- se till att alla är överens om innebörden av varje klass
  - vad är syftet med klassen?
- var noga med namngivningen
  - ska vanligtvis vara ett beskrivande substantiv i singularis

Student
Person som är inskriven vid fakulteten.

Registreringsformulär
Formulär som innehåller en students namn, personnummer, linje, antagningsår, läsår, samt val av kurser under läsåret.

CRC-kort

En standard för att översiktligt dokumentera klasser

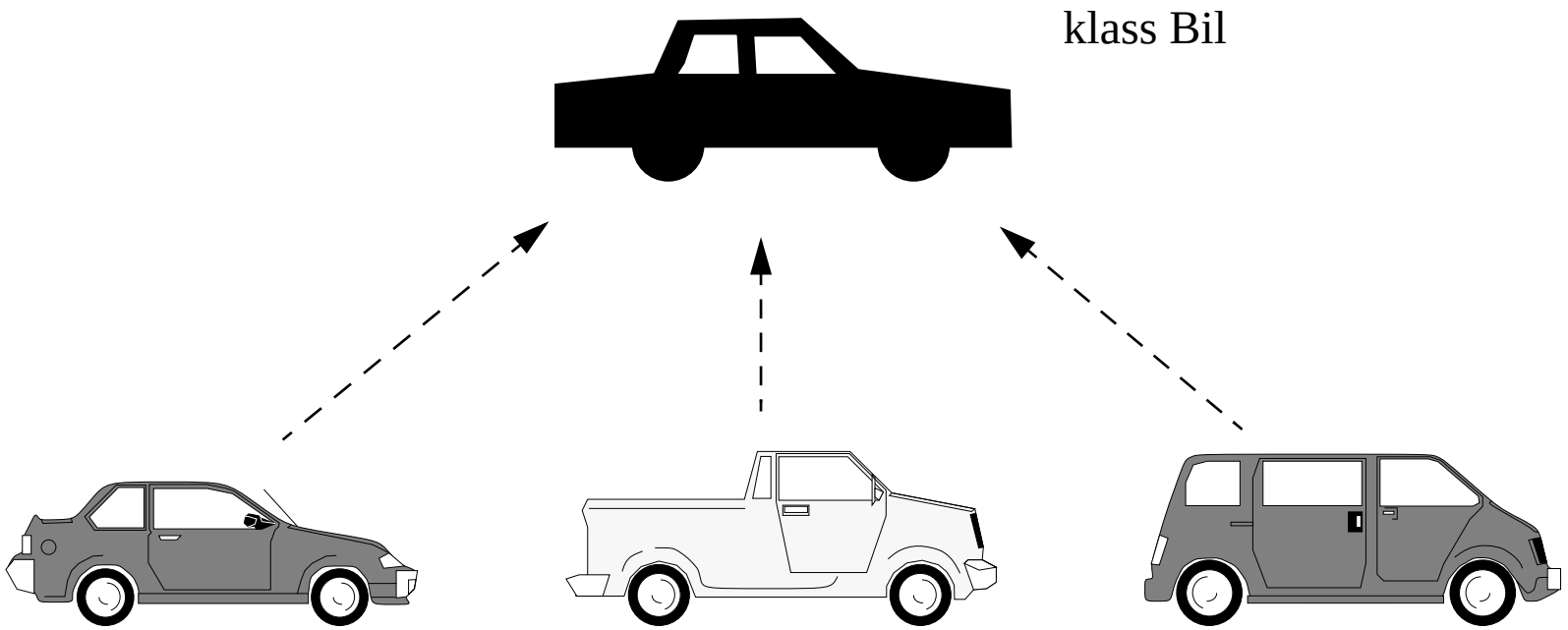
- *Class* – klassens namn
- *Responsibilities* – ansvar som klassens har (”max 3-4 stycken”)
- *Collaborators* – samarbetspartners (andra klasser)
- baksidan kan användas för att notera dataattribut

E-brevlåda	
hämta nästa brev radera brev svara på brev	Brev Brevkö

lista med inkomna brev
lista med sparade brev

Klassificering

- en klass representerar alla objekt med samma egenskaper



Klasser

- en klass beskriver det som är gemensamt för alla objekt av typen ifråga
- kan ses som en stämpel eller stans för att skapa objekt

klassnamn

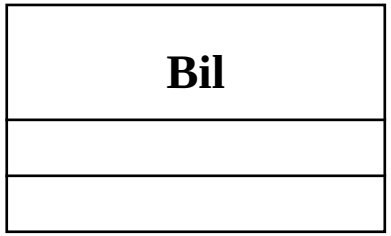
data

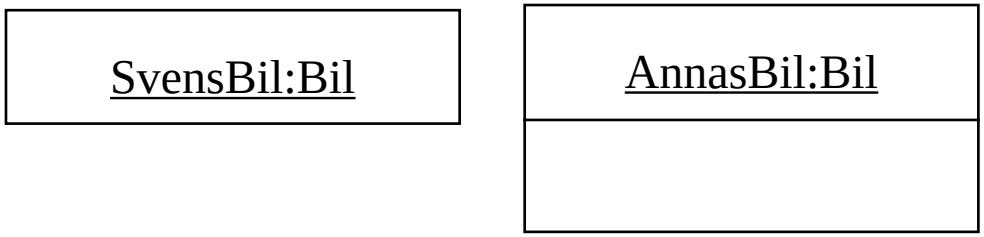
operationer

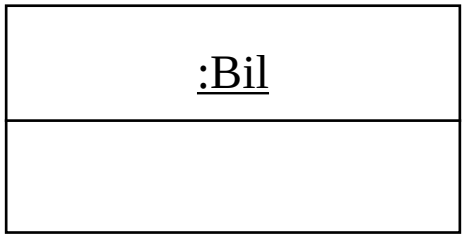
Objekt

Objekt är klassinstanser

- är konkreta förekomster av klassen
- skapas under programkörningen







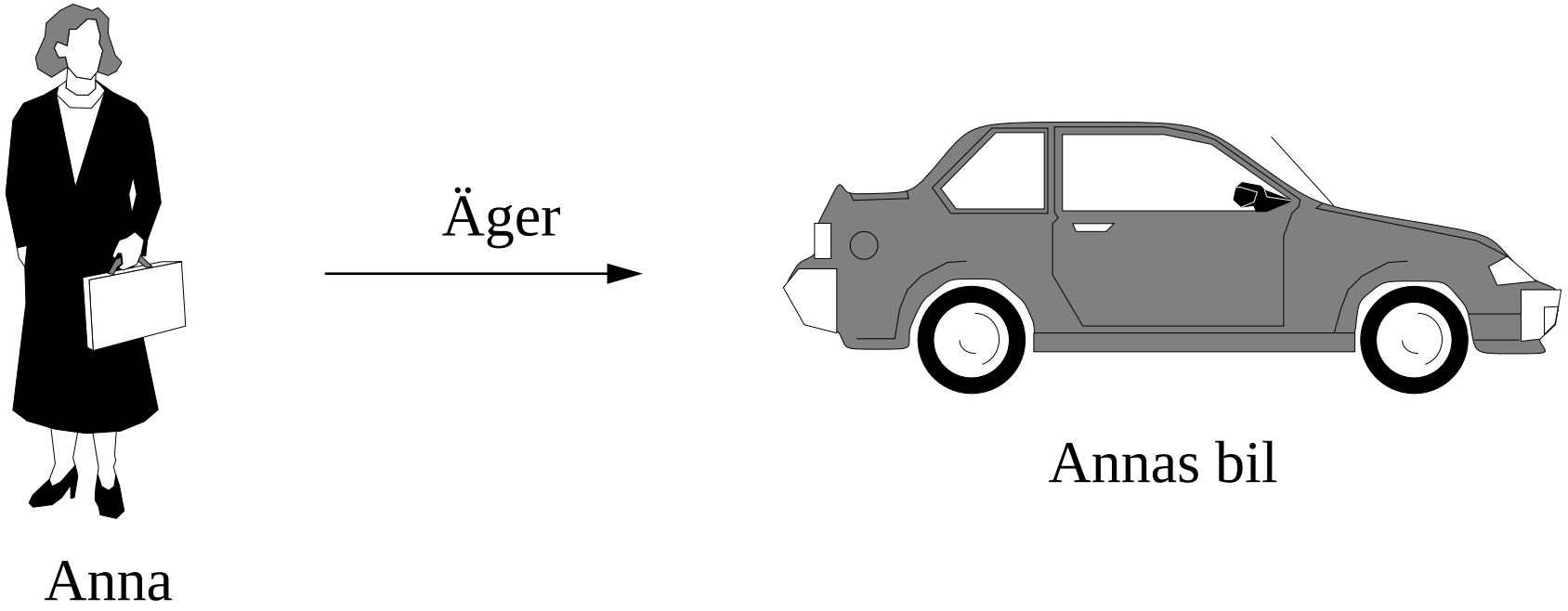
Relationer

Relationer mellan klasser eller objekt kan grovt delas upp i

- *härledning – arv*
- andra former av relationer, som
  - *association* – ”känner till”, ”har en”
  - *”aggregation”* – en ”helhet–del”-relation (en form av sammansättning)
  - *”composition”* – som ”aggregation” men livstiden för delarna kontrolleras av det ”hela”

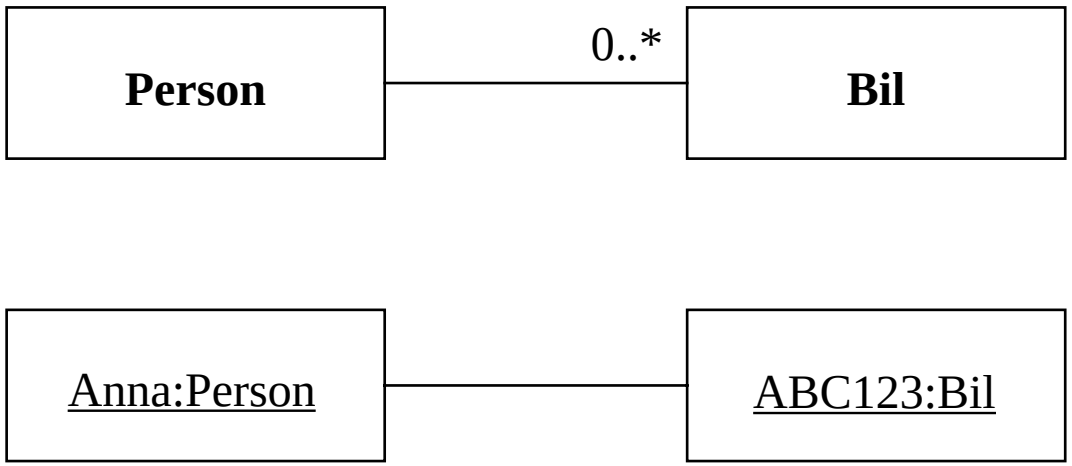
Association

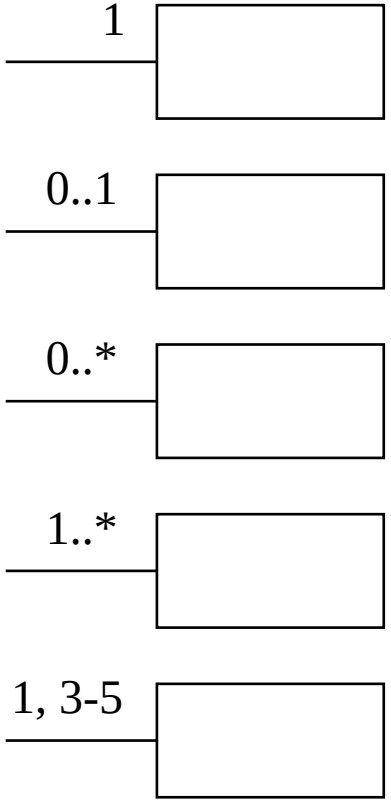
- objekt kan ”känna till” och ”använda” andra objekt i sin omgivning



Association, forts.

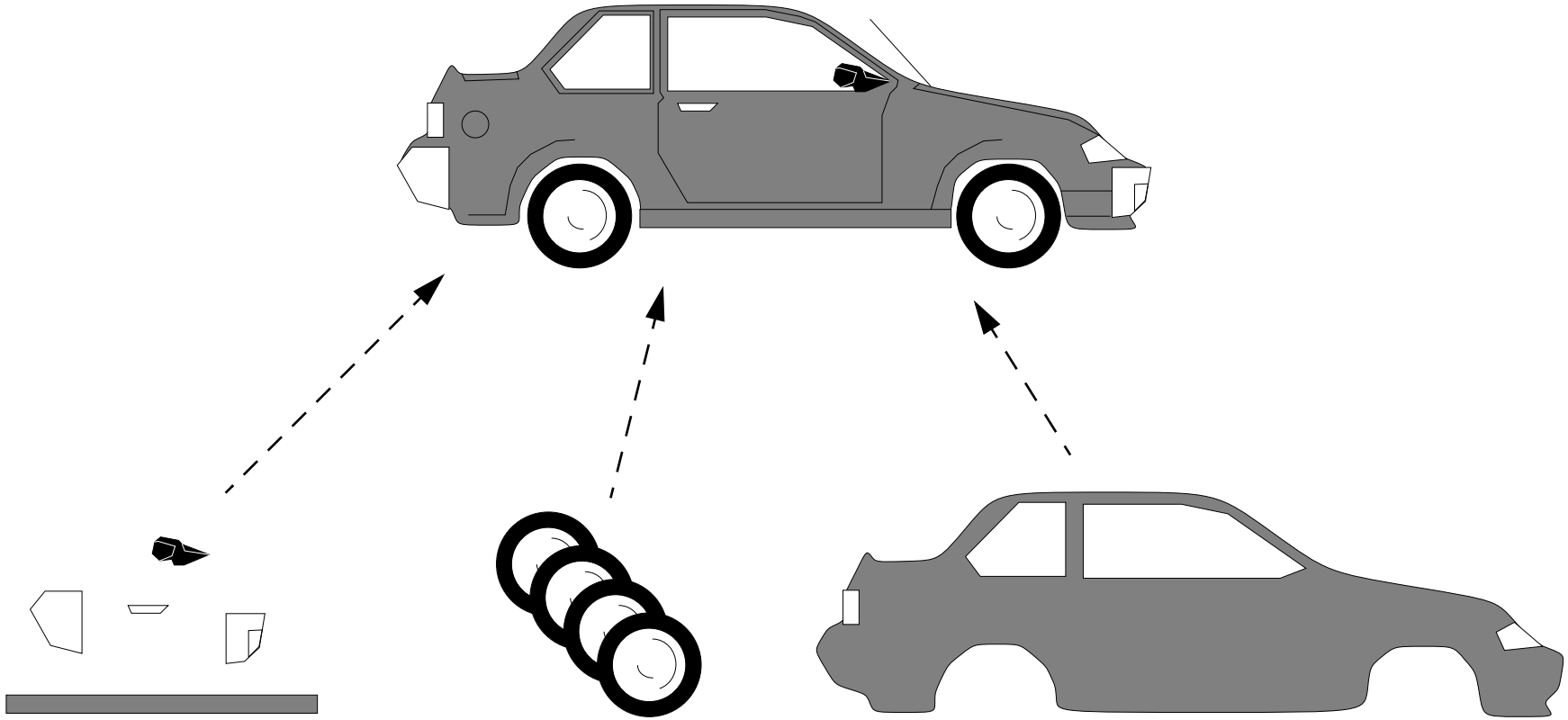
- association mellan *klasser* anger hur klassens objekt kan vara relaterade
- en association mellan *objekt* gäller specifika objekt
- multiplicitet, riktning m.m. kan anges i diagrammen





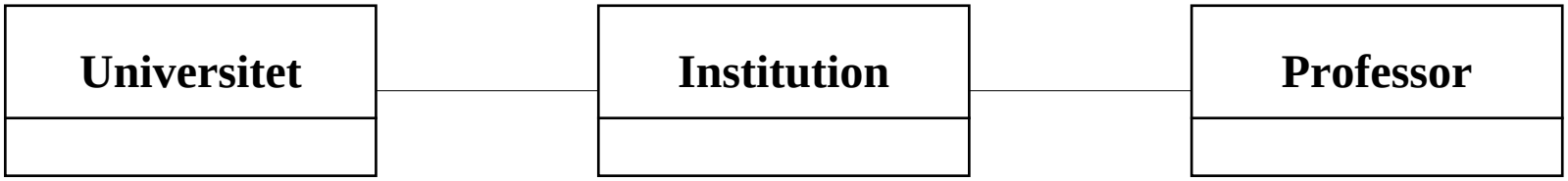
”Aggregation”

- objekt består av andra objekt – ”helhet/del”-relation
- innebär *inte* ägarskap – det behöver inte innebära att delobjekt förstörs då det ”hela” förstörs



”Composition”

- som ”aggregation” men ”composition” innebär *ägarskap*
  - det ”hela” kan ha direkt ansvar för att skapa och destruera delarna
  - det ”hela” kan tänkas acceptera en redan skapad del och senare överlämna den till en annan ”hel”



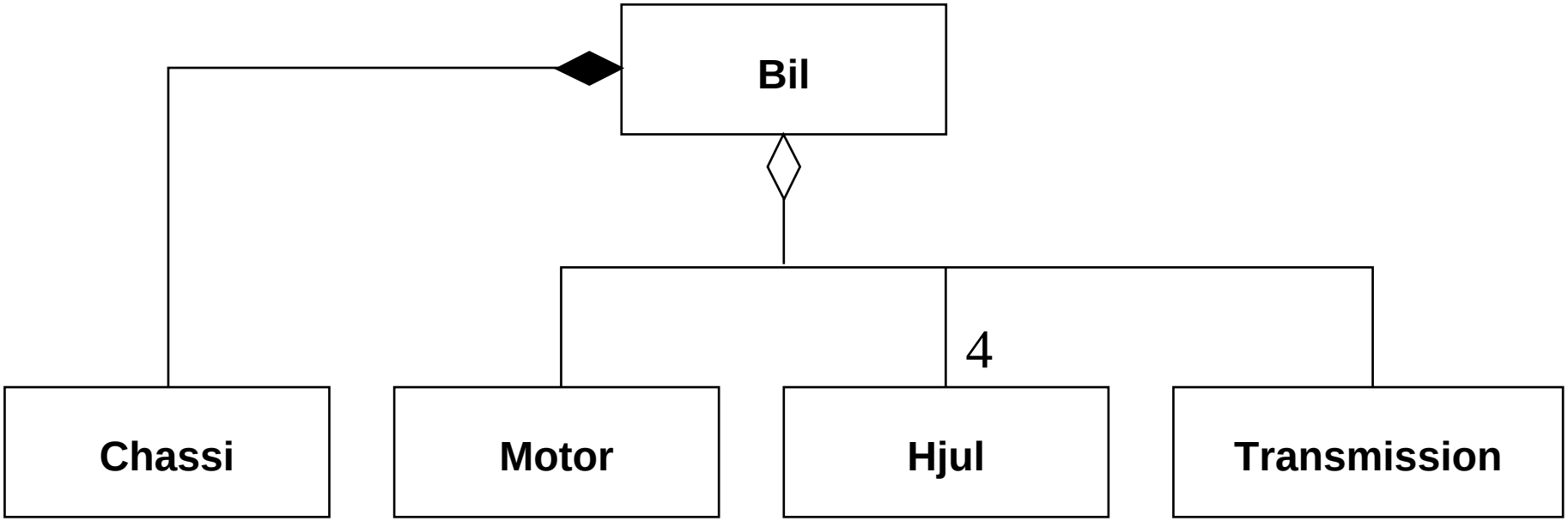
- om ett universitet läggs ner kommer dess institutioner inte längre existera (”composition”)
  - professorerna kommer att fortleva (flyttar kanske till ett annat universitet)
- om en institution läggs ned
  - professorerna flyttar kanske till en annan institution

Inte alltid lätt att bestämma vilken av association – ”aggregate” – ”composition” som bör väljas.

Implementering? Pekare – referenser – medlemsskap.

Diagram för att beskriva ”aggregate” och ”composition”

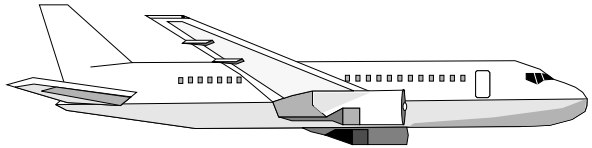
- en *rutersymbol* placeras vid det sammansatta objektet
- ofylld rutersymbol anger ”aggregation”
  - objekten behöver inte ha samma livstid som ”helheten” (Bil)
- en fylld rutersymbol anger ”composition”
  - objekten kan ha samma livstid – objekten delas inte med andra objekt

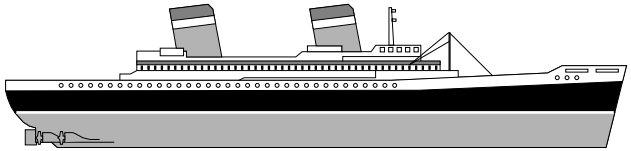


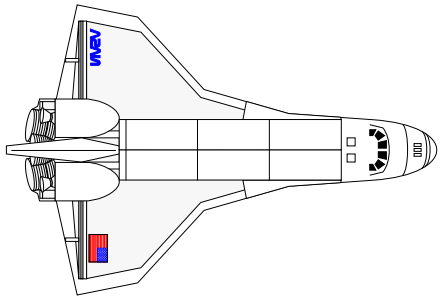
Generalisering – specialisering

- beskriver likheter mellan klasser

Vilka likheter finns?

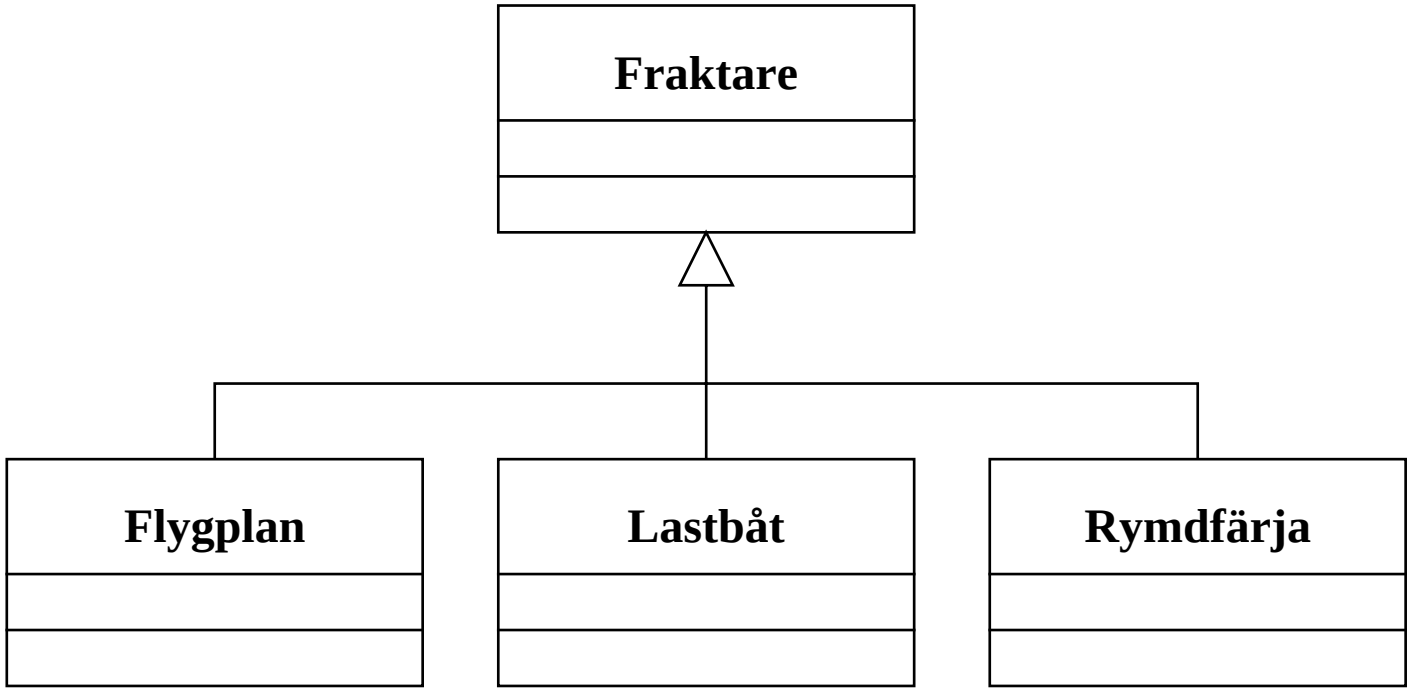






En möjlig arvshierarki för transportmedel.

- triangeln sitter vid basklassen



Dokumentering av projektet i kursen

Samtliga dokument ska bestå av följande, utöver de dokumentspecifika delarna

- försättsblad
  - projektets namn
  - typ av dokument (analys, design, …)
  - datum
  - projektmedlemmar
- innehållsförteckning
- inledning
- dokumentkonventioner (t.ex. användning av olika typsnitt, kursiv och fet stil, …)

Kravspecifikation är redan gjord

- en funktionell beskrivning av systemet
- utgångspunkt för analysen

Objektorienterad analys – OOA

Analys omfattar typiskt följande moment:

- finn objekten
  - skapa en lista med alla objekten
- klassificera objekten
  - vilka klasser ska finnas, vad ska de heta
  - använd exempelvis CRC-kort
- identifiera och utför användningsfall
  - utförs och beskrivs steg-för-steg
  - kan även komma först och då vara ett sätt att finna objekten
  - underlag för testning
- beskriv relationer mellan klasser
  - rita klassdiagram och eventuella andra diagram
- om det finns delsystem eller liknande kan det kan vara aktuellt att gruppera klasserna

En iterativ och inkrementell process!

Användningsfall

Fokus på *externt* beteende

- systemet specificeras utifrån användarens perspektiv
- bildar grund för både konstruktion och testning
- användningsfallsmodellering ger två huvudsakliga resultat
  - aktörer
  - en katalog med användningsfall
- utgör funktioner som är synlig för användaren
  - uppnår distinkta mål för användaren
- kan vara stora eller små
  - kan bestå av delfall

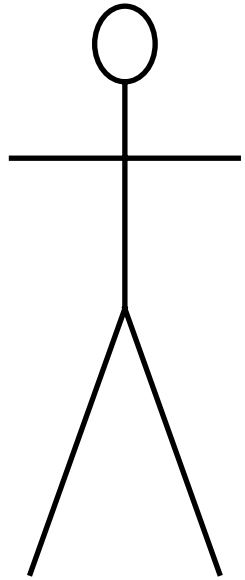
Definiera aktörer

Aktörer

- är systemexterna
- utbyter information med systemet
- används för att modellera interaktion med systemet
- analyseras inte i detalj

Skilj på *användare* och *aktör*.

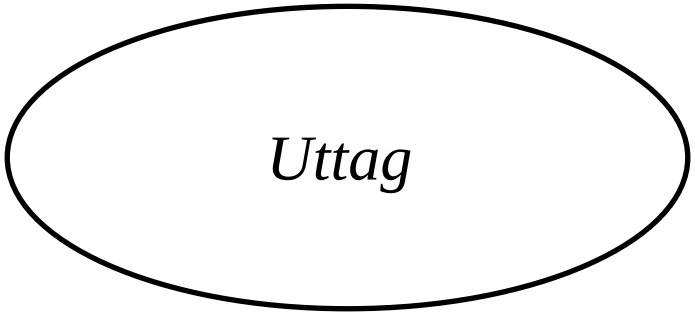
- en användare är en person eller en sak
  - herr Linus, fröken Linnéa
- en aktör är en *roll* som en användare kan anta
  - avdelningschef, bankkund



Bankkund

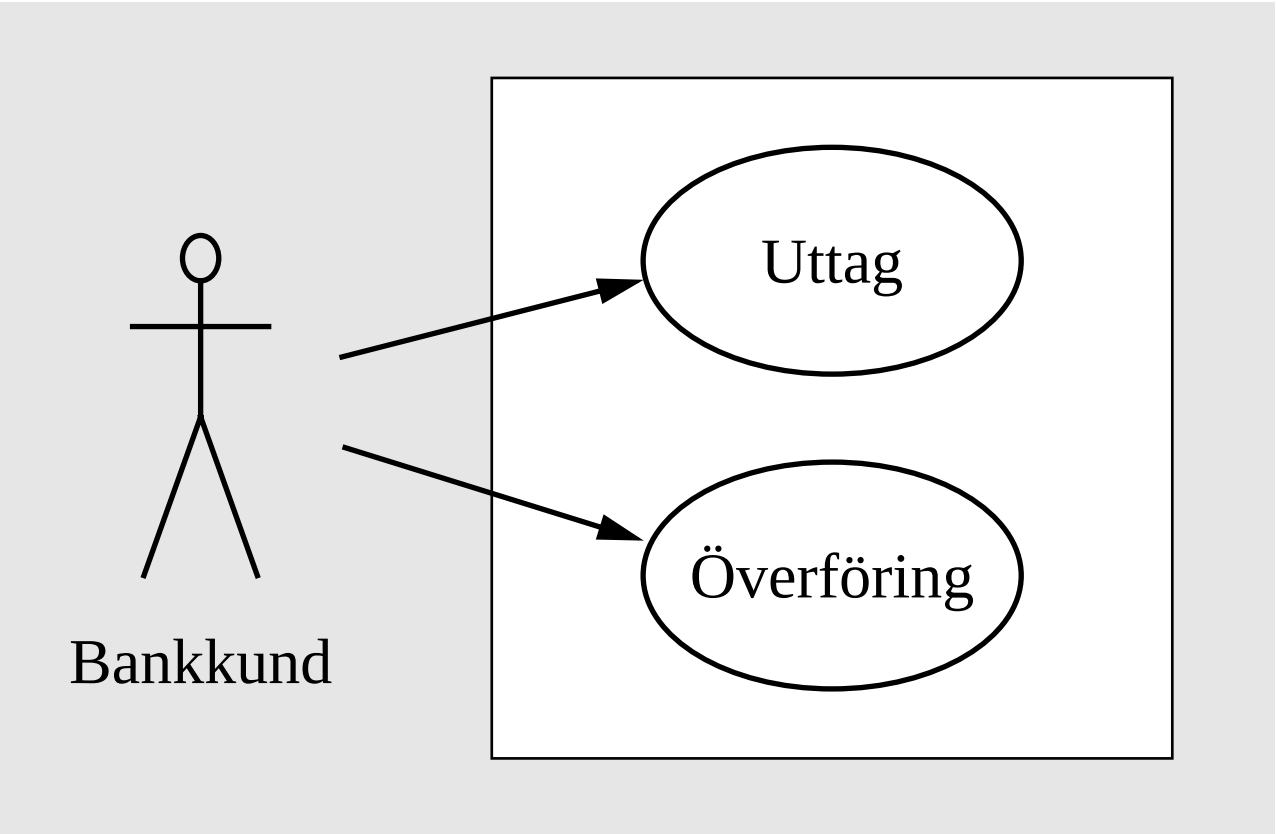
Definiera användningsfall

- utgör en sekvens av händelser i systemet
  - ger den ordning som operationer ska utföras i
- initieras av en aktör
  - bankomatkund
  - rökdetektor
- visar systemet funktionalitet
  - *vad* ska göras, hända
- användningsfall är den funktionalitet en aktör använder
  - göra uttag
  - utlösa brandlarm

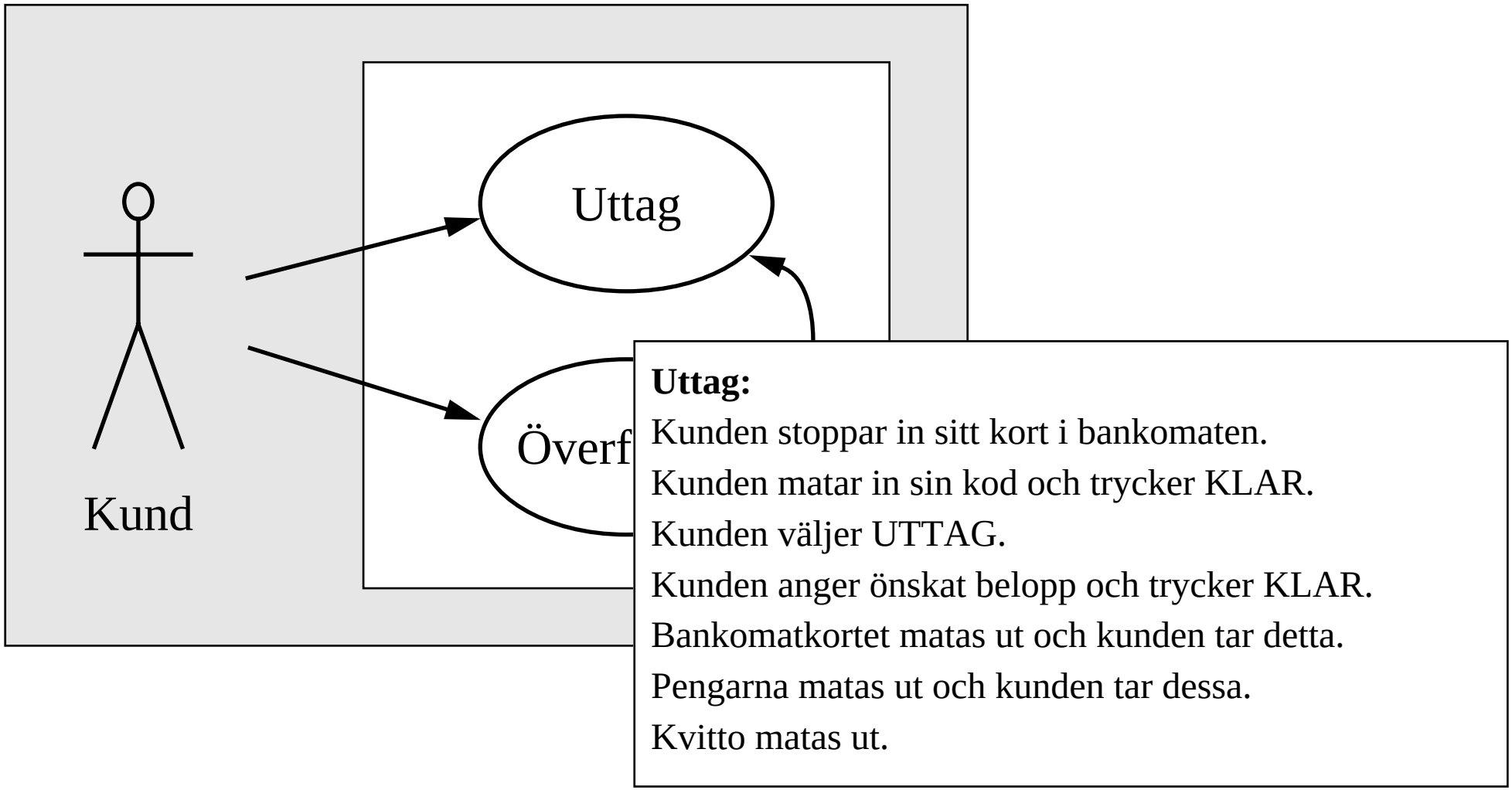


Användningsfallsmodell

- systemet i en box (den vita nedan)
- en aktör har en extern roll – befinner sig utanför boxen
- användningsfall visas som ellipser

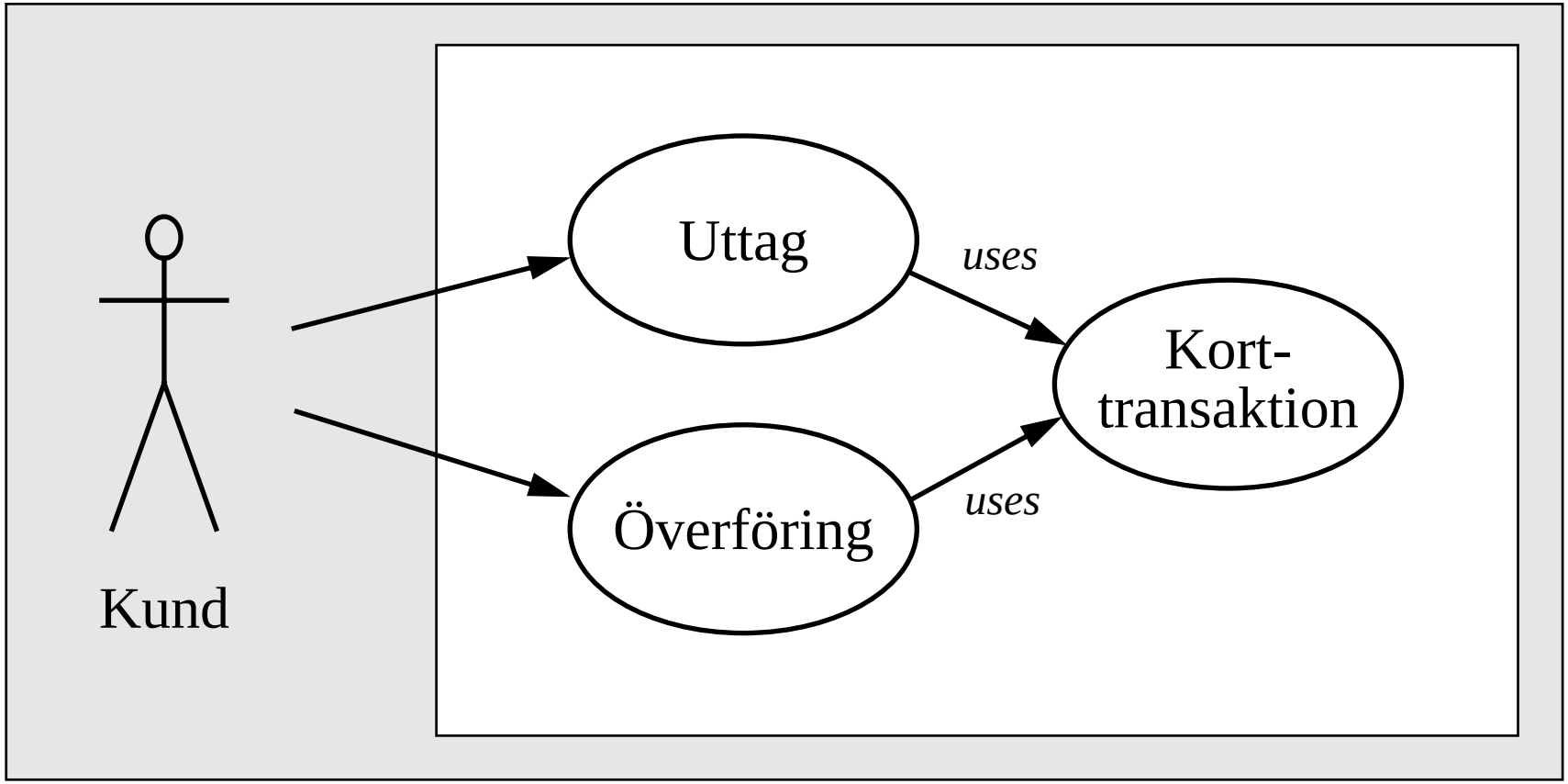


Användningsfall beskriver detaljer



Användningsfall kan använda andra användningsfall

Uses och *extends*.



Diagram

- klassdiagram
  - visar statiska samband mellan klasser och därmed samband mellan objekt
- sekvensdiagram
  - varje objekt har en egen ”livslinje”
  - pilar mellan objektens ”livslinjer” visar funktionsanrop och tidssekvens
- samarbetsdiagram (alternativ/komplement till sekvensdiagram)
  - visar relationer mellan ingående objekt
  - på sammanbindningslinjerna mellan objekten anges funktionsanropen
  - ordningen kan visas genom numrering av anropen
- aktivitetsdiagram
  - beskriver arbetsflöden
  - beskriver samtidiga förlopp eller förlopp vars inbördes ordning är oviktig

Dokumentering av analysfasen

Analysdokumentet ska omfatta följande dokumentspecifika delar

- kort allmän beskrivning av projektet och dess ramar (återanvänd från KS)
- beskrivning av användargränssnitt (återanvänd från KS)
- beskrivning av permanent datalagring, i förekommande fall (återanvänd från KS)
- klasskatalog – en översikt av alla klasser och deras syfte
- användningsfall – namn och utförande
  - om många, ska åtminstone några väsentliga fall dokumenteras mer noggrant
  - övriga kan då i princip listas med kort kommentar
- diagram av olika slag
  - klassdiagram – obligatoriskt
  - eventuellt sekvensdiagram, samarbetsdiagram, tillståndsdiagram, aktivitetsdiagram
- klassbeskrivningar, motsvarande till exempel CRC-kort med tillagda kommentarer

Utgångspunkt för designfasen.

Eventuella brister korrigeras i designdokumentet.

Objektorienterad design

- systemkonstruktion
  - systemarkitektur – användargränssnitt, permanentlagring av data, delsystem, …
  - val av klassbibliotek, standarder, etc.
- infrastrukturkonstruktion
  - till exempel klasser för ett databasgränssnitt
- detaljkonstruktion
  - klassdesign

Detaljdesign av klasser

Klassbeskrivningsformulär kan användas

- klassens namn
- klassens syfte
- eventuella basklass(er) vid arv
- konstruktor(er)
- andra medlemsfunktioner
  - typ av returvärde
  - parametrars namn och typ
  - om undantag kastas
- datamedlemmar
  - namn och typ

Dokumentering av designfasen

- uppdaterade beskrivningar och diagram från analysen
- detaljdesign för klasserna
  - motsvarande klassbeskrivningsformulär
  - får vara C++-orienterad

Utgör underlag för kodning.

Dokumentering av implementeringsfasen

Implementeringsfasen dokumenteras i form av programkoden

- all kod ska ha en enhetlig stil – projektgruppen beslutar vilken
- exempel på kodmallar finns på webben
- det ska framgå vem som gjort vad

Erfarenhetsrapport

Gruppen ska i samband med projektets avslutning lämna in en erfarenhetsrapport

- en gemensam sammanställning av gruppens samlade erfarenheter, eller
- delar bestående av varje gruppmedlems personliga erfarenheter