Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

Introduktion till UMLs klassdiagram

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

1 Inledning

Systemvaruhusets vision är att revolutionera IT-branschen genom att göra system-utvecklingsarbetet förutsägbart. För att uppnå förutsägbarhet så arbetar vi med UML som modellspråk i våra utvecklingsprojekt. För att kunna revolutionera branschen krävs det att fler än vi ändrar vårt arbetssätt. Vi delar därför med oss av vår kunskap, på webben (www.systemvaruhuset.se), genom utbildningar och i våra uppdrag som konsulter.

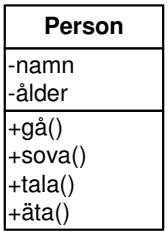
UML är ett standardiserat, generellt objektorienterat modellspråk som ägs och förvaltas av Object Management Group (OMG). Det består av 13 diagram-typer. Detta dokument ger en introduktion till klassdiagrammet. Det är viktigt att komma ihåg att UML inte säger något om hur man bygger bra system utan enbart hjälper oss att beskriva dessa.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

2 Klassdiagram – Introduktion till klasser och objekt

Vid objektorienterad analys och design så används klassen för att representera en företeelse. Denna företeelse kan uppvisa vissa egenskaper (attribut och relationer) och ett visst beteende (operationer). Företeelsen kan dessutom förekomma ett otal gånger (instanser av klassen) i ett system.

Företeelsen kan vara något som finns i verkligheten, såväl något konkret (t.ex. bilar och båtar) som något abstrakt (t.ex. kärlek och hat). Företeelsen kan också sakna motsvarighet och i stället vara en logisk konstruktion (såsom program-språksspecifika företeelser som ”array” och ”hashtable”). Formellt så är en klass en egendefinierad datatyp i ett objektorienterat system.



En klass kan liknas vid en mall, form eller ritning som definierar de egenskaper och det beteende som alla förekomster (objekt) som finns av en viss klass. Ofta liknas en klass vid en ritning och ett objekt vid det specifika huset som byggts enligt ritningen. Varje förekomst har sina egna attributvärden. Alla hus har en adress, men varje hus har sin egen unika adress (vasagatan 1, drottninggatan 2 etc.).

Vid objektorienterad systemutveckling utgör systemets samtliga klassbeskrivningar tillsammans systemets källkod. Objekten skapas sedan i runtime-miljön.

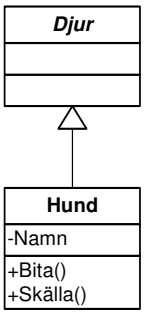
En klass antas alltid vara en del av det paket den visas inuti. Vill man visa en referens till en klass från ett annat paket kan man specificera sökvägen till denna i namnsektionen enligt formatet *Klassnamn::Paketnamn.* Paketnamnen avskiljs alltså med dubbla kolon (::)

UML-not: Med klassificerare avses en klass från UMLs metamodell. Närmare bestämt superklassen (föräldern) till klass, datatyp och gränssnitt. Samtliga av dessa subklasser/-specialiseringar visas med hjälp av klassrektanglar. Datatyper och gränssnitten kompletteras dock med nyckelorden (*DataType* och *Interface)*

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

2.1 Olika typer av klasser

2.1.1 Abstrakta klasser



Vissa klasser kan man ej skapa objekt av (instansiera). Dessa kallas för abstrakta klasser och används för att man skall kunna återanvända delar (operationer, attribut, associationer) av dessa klasser genom arv/specialisering till subklasser. Subklasserna kan göras konkreta, d.v.s. de kan användas för att skapa objekt av. Formellt så är inte abstrakta klasser en egen typ utan en vanlig klass med vissa begränsningar.

Abstrakta klasser visas genom nyckelordet *{abstract}* och/eller genom att klassens namn skrivs med kursivstil (*Klassnamn*). Samma konvention, kursiv stil i namn, används för att visa på abstrakta operationer (d.v.s. operationer utan implementation). Grundprincipen är att om en klass har någon abstrakt operation så måste hela klassen definieras som abstrakt.

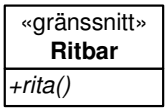
2.1.2 Gränssnitt (eng. Interface)

Ett gränssnitt specificerar de externt synliga operationerna i en klass, komponent eller annan klassificerare (inklusive delsystem) utan att specificera hur dessa realiseras (klassens interna struktur).

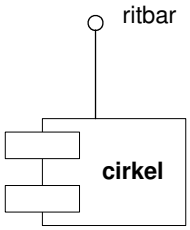
En klass kan specificera flera olika gränssnitt eftersom ett gränssnitt ofta enbart definierar en delmängd av klassen totala funktionalitet. Det kan också vara så att ett och samma gränssnitt implementeras av flera olika klasser. Det klassiska exemplet på detta är gränssnittet ”ritbar” som specificerar operationen ”rita”. Detta gränssnitt implementeras av klasserna ”Cirkel”, ”Triangel” och ”Rektangel”.

Gränssnitt kan generaliseras och specialiseras men skiljer sig från en abstrakt klass på så sätt att gränssnittet varken har några attribut eller tillstånd (enbart operationer, som inte är implementerade). En klass beter sig gentemot ett gränssnitt i en speciell roll (utnyttjar ett visst gränssnitt).

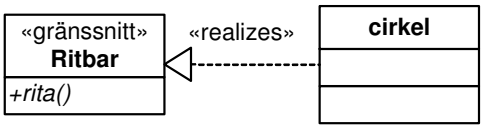
Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se



Ett gränssnitt visas med hjälp av en klassrektangel med nyckelordet «interface». Listan med gränssnittets operationer placeras då i operationssektionen medans attributsektionen kan utelämnas eftersom den per definition alltid är tom. Ett annat sätt att visa ett gränssnitt på är som en liten cirkel med namnet placerat nedanför symbolen. Cirkeln kan kopplas till en klass som realiserar den med hjälp av en heldragen linje. Detta visar att den påkopplade klassen tillhandahåller minst de operationer som finns i gränssnittet. Operationerna namnges dock ej vid cirkeln.



En klass som använder sig av operationerna i ett gränssnitt sägs vara beroende av dessa. Det kan visas med en streckad pil som pekar på cirkeln. Klienten behöver inte vara beroende av gränssnittets alla operationer. Beroendet kan också visas med en klo (en heldragen linje som avslutas med en öppen halvcirkel) som omsluter gränssnittssymbolen.



Realiseringsrelationen mellan en klassificerare och ett gränssnitt som stöds kan också visas med hjälp av en streckad linje med ett solitt triangulärt pilhuvud ("en streckad arvsrelation").

2.1.3 Datatyper

En datatyp skiljer sig från en "vanlig" klass genom att den identifieras enbart med hjälp av sina attribut. Den har alltså ingen identitet i samma mening som övriga klasser. I normalfallet används datatyper för att beskriva primitiva typer i systemet (t.ex. integer och string). En datatyp kan ha egna attribut. Dessa är då också primitiva typer.

2.2 Klassdiagrammets användningsområde

Klassdiagrammet används för att strukturera företeelser och deras inbördes relationer, d.v.s. att beskriva strukturer. Det spelar egentligen ingen roll vilken typ av struktur det rör sig om. Det gör att klassdiagrammet kan förekomma i alla faser av såväl verksamhets- som systemutveckling. Den teoretiska definitionen av

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

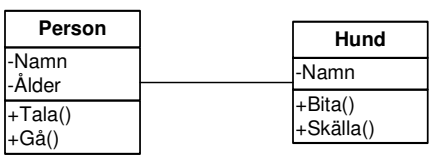
ett klassdiagram är att det är en graf med klassificerare (t.ex. klasser med attribut och operationer) som är sammankopplade med olika statiska relationer (t.ex. associationer och generaliseringar). Klassdiagrammet kan sägas bestå av en samling statiska, deklarativa modellelement (klasser, gränssnitt etc.) och kan organiseras i olika paket.

Exempel på användningsområden för klassdiagrammet är bl.a. för att dokumentera målmodeller (målstruktur), begreppsmodeller (begreppsstruktur) och informationsmodeller (informationsstruktur) samt analys och designmodeller (systemstruktur). Diagrammet är ryggraden vid analysarbete i allmänhet och objektorienterad systemutveckling i synnerhet.

Klassdiagram används under nästan hela utvecklingsprojektet och detaljeringsgraden förfinas succesivt. Ett klassdiagram kan innehålla gränssnitt (eng. Interfaces) paket, relationer och till och med objekt med länkar. Därför används ibland begreppet statiska strukturella diagram istället för klassdiagram.

Ett system kan presenteras uppdelat på flera olika klassdiagram. Denna uppdelning betyder inte något i sig. Den görs oftast av praktiska skäl (för att öka överskådligheten). I UML är det viktigt att hålla isär begreppen diagram och modell. Modellen beskriver helheten medans ett diagram beskriver ett utsnitt av modellen, dvs klassmodellen beskriver samtliga klasser i systemet medans klassdiagrammet beskriver ett antal utvalda klasser. Det är dock givetvis möjligt, om än inte alltid så praktiskt, att skapa klassdiagram som innehåller alla klasser.

2.3 Notation – Att rita klassdiagram



En klass visas som en rektangel med heldragna kantlinjer. Rektangeln har tre sektioner som avdelas med hjälp av horisontella linjer. Den översta sektionen innehåller klassens namn och andra generella egenskaper. Mittensektionen innehåller en lista med klassens attribut och den nedre sektionen innehåller listan med operationer.

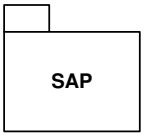
Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

Det är valfritt att visa såväl attributsektionen som operationssektionen. Olika verktyg kan även stödja ytterligare sektioner.

Notationen för objekt är likadan som den för klasser med den skillnaden att den övre sektionen innehåller såväl objektets namn som klass. Namnet är dessutom understryket. Objektnamnet särskiljs ifrån klassnamnet med hjälp av ett kolon (:). Fullständiga sökvägar till objekt i andra paket kan även visas, paketen avskiljs då med dubbla kolon (::). *objektnamn:Klassnamn* eller *objektnamn:Klassnamn::paketnamn*

2.4 Att strukturera klassdiagrammet med hjälp av paket

Är klassdiagrammet komplext och innehåller många klasser kan dessa delas upp i flera paket. Ett paket är en gruppering av modellelement. Ett paket kan dels innehålla modellelement såsom klasser, dels andra paket. Strukturen kan på detta sätt bli hur djup som helst. Alla typer av UMLs modellelement kan grupperas i paket. Paketen används för att skapa en övergripande struktur som gör det lättare att läsa, förstå och navigera i strukturen. Det finns alltid ett ”yttersta” paket.



Ett paket visas som en stor rektangel med en liten rektangel (en flik) fastsatt ovanpå till vänster på den stora. Paketets innehåll kan antingen visas inuti den stora rektangeln eller så ritas det som egna paket utanför (vanligen nedanför) det omgivande paketet men kopplas samman med en heldragen linje och ett plustecken (+) inuti en cirkel sätts närmast det omgivande paketet. Visas inte paketets innehåll i den stora rektangeln kan namnet skrivas där, i annat fall skrivs det i den lilla rektangeln.

Ett nyckelord kan skrivas in ovanför paketets namn. UMLs fördefinierade stereotyper är; *facade, framework, stub,* och *topLevel.* Egenskaper kan också placeras inom ”måsvingar” efter eller under paketets namn (t.ex. *{abstract}*). Paketets synlighetsgrad visas med samma symboler som gäller för operationer och attribut (+, -, # och ~) och skrivs framför paketets namn.

Not: Observera att paketen äger modellelementen som ligger inuti respektive paket och att dessa är grunden för konfigurationskontroll, åtkomst och förvaring. Varje element kan bara ägas direkt av ett paket. Det är dock möjligt att referera

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

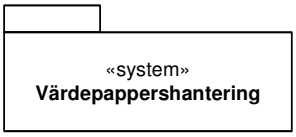
till element i andra paket. Detta visas i modellen med stereotyper som «import» och «access».

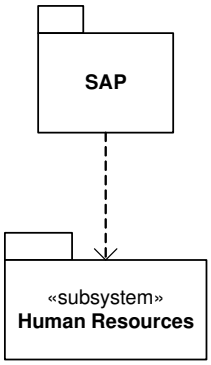
Det är inte nödvändigt att visa hela paketets innehåll inuti rektangeln. Det kan mycket väl vara en delmängd som väljs utifrån något kriterium. Detta är en generell regel som är applicerbar för samtliga UML-diagram.

2.5 Relationer mellan paket

UML har definierat tre typer av relationer som kan användas mellan paket (fyra om man räknar med merge, men det är ett specialfall av beroende som framför allt används i paketdiagrammet). Relationer mellan paketen kan visas med hjälp av linjer mellan paketsymbolerna.

- Association
- Beroende
- Arv





Det kan finnas beroenden mellan olika paket. Det beroende paketet kallas ofta klient medan det andra kallas server. Ett paket som är beroende av andra innebär ofta att klientpaketet behöver anropa operationer i serverpaketet. Det betyder dels att klientpaketet behöver känna till att serverpaketet finns (och vart) samt att förändringar i serverpaketet kan påverka klientpaketet. Vid objektorienterad analys och design strävar man ofta efter enkelriktade pilar.

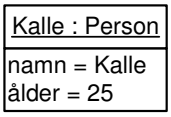
Ett paket kan ärva från andra paket. Det ärvande paketet ärver då ett interface/gränssnitt.

Import och accessrelationer visas med hjälp av en streckad pil med öppet pilhuvud med nyckelorden «import» eller «access», vid sin sida. Element från importerade paket kan antingen visas utanför paketsymbolen eller inuti densamma. Paket med omfattande innehåll visas vanligen som enkla ikoner med namn som användaren sedan kan klicka sig vidare ned i på egen hand.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

2.6 Objekt

Ett objekt representerar en specifik instans (förekomst) av en klass. Objektet har en identitet och attributvärden. När man skapar ett objekt av en viss klass säger man att man skapar instanser av ett objekt (instansierar objekt) av en viss typ/klass.



Objektets namn kan ibland utelämnas (t.ex. vid anonyma objekt). Det är dock viktigt att i dessa fall lämna kvar kolonet vid klassnamnet och stryka under detta. Ett anonymt objekt får sin identitet utifrån sina associationer. Liksom objektets namn kan även dess klass utelämnas (dock ej samtidigt som namnet...). Utelämnas klassnamnet utelämnas även kolonet.

Objektnamn kan utelämnas för att visa på anonyma objekt från en viss klass (":KlassNamn"). Klassnamn kan utelämnas när det framgår av sammanhanget. ("objektNamn"). Attributsektionen är valfri att visa.

Vill man visa att ett objekt är en instans av flera olika klasser används en kommaseparerad lista med klassnamn. Detta kan enbart implementeras i programspråk som stödjer multipelt arv. Objekt kan visas inuti olika typer av diagram, bl.a. klassdiagrammet.

Not: En notation som liknar objektets används för att visa på roller, t.ex. i ett kommunikationsdiagram. Notationen för en roll i en samverkan (**eng. Collaboration**) liknar den som används för objektet eftersom roller har instansliknande egenskaper. En roll kan sägas vara en instans (eller klass) i ett visst sammanhang. Det kan dock finnas många olika typer av instanser (klasser) som kan spela denna roll eftersom det man i samverkansdiagrammet vill visa på är det ansvar en roll har, inte vilken typ den har.

3 Attribut och operationer

Attribut och operationer används för att specificera egenskaper och beteende på analysnivå och data respektive metoder på designnivå.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

3.1 Vem kan anropa vem – synlighetsgraden för attribut och operationer

Synlighetsgraden används för att definiera vilka av ett objekts attribut respektive operationer som är synliga för andra objekt. Det finns fyra olika synlighetsgrader (eng. Visibility):

+ publika (eng. Public) attribut

skyddade (eng. Protected) attribut

- privata (eng. Private) attribut

~ paket (eng. Package) attribut

Publika attribut/operationer kan ses av alla objekt i systemet, skyddade attribut kan ses av alla objekt i en och samma arvshierarki, privata attribut kan enbart ses av det egna objektet och avslutsningsvis så kan attribut definieras med en synlighet på paketnivå. Det innebär att attributen är synliga för alla objekt från samma paket.

Synlighetsgraden kan utelämnas. Det är dock inte samma sak som att den är odefinierad eller att attributet är publikt. Synlighetsgraden kan även visas med hjälp av nyckelord (*public, protected, private, package*).

Ytterligare typer av synlighetsgrader kan definieras av användaren. Alla synlighetsgrader förutom publik är implementationsspecifika, d.v.s. implementationen beror på vilket programspråk man använder.

Attribut och operationer kan på detta sätt gömmas från andra objekt genom att användaren specificerar en annan synlighetsgrad än ”publik”. Det görs för att åstadkomma en bättre inkapsling. Den grundläggande designprincipen är att så lite som möjligt av klassen skall vara synligt utåt. Attributs- och operationssektionerna kan även utelämnas från diagrammet för att öka läsbarheten.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

3.2 Attribut

Textsträngar i attributsektionen (mitten) används för att visa klassens attribut. Dessa representerar objektets egenskaper. En liknande syntax används för att visa kvalificerare, mallparametrar, operationsparametrar etc.

Attributen definieras enligt följande syntax: *synlighetsgrad* : *typ* [multiplicitetsordning] = *startvärde* {*egenskapsvärden*} – public AttributNamn: Typ = Startvärde.

Atributet "+string:belopp [0..* ordered]=0 {frozen}" är alltså ett attribut med namnet "belopp" som är synligt för alla andra objekt (+), det kan finnas flera samtidiga belopp (0..*), men dessa är åtminstone ordnade på något finurligt sätt (ordered). Attributets startvärde är 0 (=0) och detta går inte att ändra (frozen).

Attributets typ kan utelämnas. I de fall ett attributs värde ändras till ett annat kan detta visas i ett objektdiagram med hjälp av en streckad pil mellan två objektsymboler med nyckelordet «become» vid pilen. Orsaken till att man kan visualisera attributets förändring i ett objektdiagram är helt enkelt att i ett objektorienterat system så är allting (eller åtminstone nästan allting) objekt, så även attribut. Skillnaden mellan ett attribut och en klass blir därför lätt en teoretisk diskussion. Ofta väljer man att lägga sådan information som inte har något eget existensberättigande som attribut istället för en klass. Det finns dock inga regler, inga "rätt" och "fel" angående detta.

Mittensektionen (med attributen) behöver ej visas, och i de fall den visas behöver ej samtliga attribut visas. Om vissa attributvärden förändras över tiden kan det visas med hjälp av en lista med de olika värdena (alternativt kan flera olika objektsymboler tillsammans med nyckelordet «become» användas enligt ovan). Möjligheterna att dölja och visa olika delar av symbolerna styrs av det verktyg man använder.

Namn på attribut inleds oftast med liten bokstav och skrivs med normal stil. Ett attribut kan ha en egen struktur, t.ex. kan attributet "Namn" bestå av ett förnamn och ett efternamn. Se även ovanstående diskussion om skillnaden mellan klasser och attribut.

http://www.systemvaruhuset.se/kunskapsbank.aspx

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

Multiplicitetsordning visar hur attributets värden arrangeras i de fall multipliciteten ej är 1..1. Följande värden kan anges:

- *{absent}* — osorterade värden
- unordered — osorterade värden
- ordered — värdena sorteras

Med *typ* avses antingen ett namn på en klassificerare eller en befintlig datatyp i det valda programspråket.

Med *startvärde* avses ett språkberoende uttryck för attributets värde hos ett nyskapat objekt. Detta värde är valfritt.

Med *egenskapsvärden* avses värden som är applicerbara på elementen. Dessa är valfria att specificera. Ett attribut som inte går att förändra visas t.ex. med hjälp av egenskapsvärdet *{frozen}*. Det finns ingen symbol som används för att visa att ett objekt faktiskt går att förändra eftersom detta är normalfallet.

Anges ingen multiplicitet har ett attribut enbart 1 värde. Multiplicitet kan visas inom hakparanteser. T.ex: *colors : Color [3]* eller *points : Point [2..* ordered]*

Observera att en multiplicitet på 0..1 betyder att värden kan saknas. Eventuella stereotypade nyckelord inom guillemet skrivs in före attributsträngen (inklusive synlighetsgraden). Egenskapsvärden följer efter strängen.

3.3 Operationer

Operationerna representerar objektens beteende och definieras med parametrar och returvärde. Operationens fullständiga syntax är: *synlighetsgrad operationsnamn {parameterlista}: returtyp {egenskaper}*. Synlighetsgraden anger vilka andra objekt som kan anropa operationen i fråga.

Exempel: + taUtPengar (int:belopp=100, string:orsak=”bio”): saldo {concurrency=guarded}.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

Parameterlistan är en kommaseparerad lista med argument till operationen. Varje parameter kan beskrivas tillsammans med sitt namn, sin typ och ett normalvärde. Listan med argument och returtypen behöver ej visas i klassdiagrammet.

Returtypen definierar den implementationsspecifika datatypen som returneras vid ett operationsanrop. Detta utelämnas av naturliga skäl om/när operationen ej lämnar ett värde i retur. Egenskaper är applicerbara på elementet. En operation som inte på något sätt förändrar systemets tillstånd kan visas med hjälp av egenskapen *{query}*. Samtidighet kan visas med hjälp av egenskapen {*concurrency = sequential*}, {*concurrency = guarded*} eller {*concurrency = concurrent*}. Är ej denna egenskap definierad antas ingen samtidighet förekomma (*sequential*).

Om en klass enbart definierar operationens signatur (namn, returvärde, argument) och inte implementerar denna måste operationen deklareras som abstrakt. Detta visas antingen med hjälp av nyckelordet {*abstract*} eller genom att operationens signatur skrivs med kursiv stil. Om en eller flera metoder i en klass definieras som abstrakta så måste även klassen som sådan definieras som abstrakt.

Text och algoritmer som beskriver operationens implementation kan dokumenteras i en anteckning/not som kopplas till operationens signatur i klassrektangeln. Beskrivs implementationen formellt (i det valda programspråket) bör den skrivas inom ”måsvingar” (*{}*) för att visa på att det är en begränsning (av utvecklarens frihet…). I annat fall görs beskrivningen som vanlig text. En implementerad operation kallas metod.

Om instanser av en viss klass accepterar och svarar på vissa givna signaler visas detta med nyckelordet «signal» vid operationsnamnet. Systemets svar på mottagandet av en signal visas enklast med hjälp av en tillståndsmaskin. Tillståndsmaskiner beskrivs med hjälp av tillståndsdiagram.

Eventuella stereotypade nyckelord (inom guillemet/måsvingar {}) placeras före operationen inklusive dess synlighetsgrad. Eventuella egenskaper beskrivs efter operationen.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

3.4 Globala attribut och operationer (Static)

Globala attribut och operationer kan anropas utan att ett objekt behöver instansieras. En klassoperation/-attribut (global operation) visas genom att namnet stryks under. Notationen är samma som för objekt. Det beror på att ett klassattribut/-operation kan jämställas med ett objekt. , med den begränsningen att det kan bara finnas ett ”objekt” (ett globalt attribut vid namn ”antalGjordaMållAllsvenskan2007”. Notationen för dessa är därför likadan. Instansoperationer är normalfallet och visas därför inte på något särskilt sätt.

4 Konventioner

Följande konventioner används i samband med klassrektangeln:

- Klassnamnet centreras, skrivs med fetstil och inleds med stor bokstav.
- Nyckelord (inklusive namn på stereotyper) visas ovanför klassnamnet och centreras också men skrivs med normal tjocklek samt omgärdas av guillemets (<< >>).
- Attribut och operationer vänsterjusteras och skrivs med normal stil.
- Abstrakta klasser och operationer skrivs med kursiv stil.
- Namn på objekt, attribut och operationer skrivs med små bokstäver.

Observera att detta är allmänt accepterade konventioner som bl.a. används i UML. Det innebär att många verktyg automatiskt använder sig av dessa, men det är inte förbjudet att göra på andra sätt.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

5 Relationer

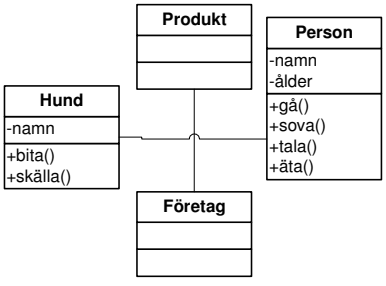
Det finns tre typer av relationer som kan användas i ett klassdiagram; associationer, beroenden och generaliseringar.

5.1 Associationer

En association deklarerar en möjlig koppling (länk) mellan instanserna av de associerade klassifierarna (t.ex. Klasser). Den består av minst två associationsändar som var och en specificerar vissa egenskaper som associationen måste inneha för att relationen skall vara giltig.

Multiplicitet är en egenskap som berättar hur många instanser av klassificeraren vid den ena änden som kan associeras en instans av klassificeraren vid den andra änden. Multipliciteten kan alltså inte vara negativ. Associationsänden informerar även om i vilken riktning kopplingen kan användas samt om vissa begränsningar (om en instans kan ersättas med en annan, om en länk kan förändras efter det att den skapats, om nya länkar kan läggas till men inga kan tas bort eller förändras). Dessa begränsningar gäller enbart länkarna, inte objekten själva.

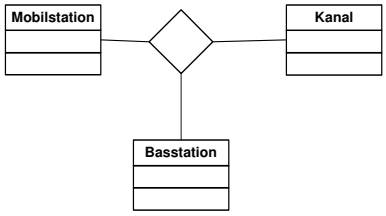
Associationer används för att visa att objekt av vissa typer kan känna till varandra. Mellan objekt finns det länkar. Mellan klasser beskrivs länkarna som associationer. En länk kopplar samman två objekt. En association är en sammanfattning av hur länkar kan existera mellan olika objekt.



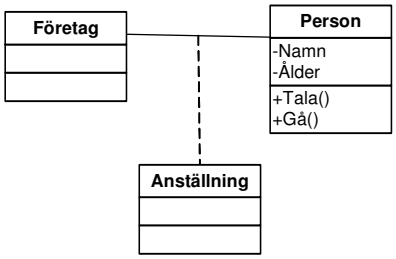
Binära associationer visas med hjälp av heldragna linjer som kopplar samman två klassificerare. Med binära associationer avses en association mellan exakt två klassificerare (inklusive rekursiva associationer, d.v.s. associationer som går till och från en och samma klass). När två associationer korsar varandra kan små halvcirkelformade ”hopp” användas för att visa att de ej är sammankopplade. Linjerna kan ha diverse utsmyckningar som visar olika egenskaper.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

Tertiära och n-ära associationer visas som diamenter med heldragna linjer till de olika klasserna som skall associeras till varandra. En association kan tilldelas ett associationsnamn och/eller en associationsklass. Vid denna typ av associationer så innebär multiplicteten 1 att minst en länk måste finnas för varje kombination av värden på de andra associationsändarna.



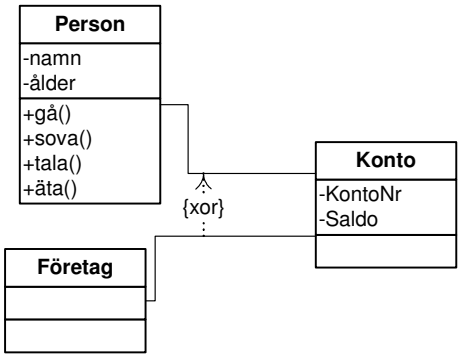
Associationsklassen visar på en association som har klassliknande egenskaper såsom attribut, operationer och andra associationer.



Associationsklassen visas med hjälp av en klassymbol som kopplas till associationen med en streckad linje. Associationen och associationsklassen är en enhet även om de faktiskt ritas och presenteras så att det ser ut som om associationsklassen är separat från associationen. Det innebär också att man inte kan specificera någon multiplicitet på associationsklassen.Alla programspråk har inte stöd för att implementera associationsklasser. Notationen används för att visa på information som är kopplad till relationen mellan två klasser, snarare än till en viss klass.

5.1.1 Xor-association

En Xor-association är en fördefinierad begränsning (eng. Constraint). Den används för att visa att bara en av flera möjliga assocationer kan instansieras i taget. Detta visas med hjälp av en streckad linje som sammankopplar de alternativa associationer och nyckelordet {xor}. Associationerna som samman-kopplas måsta alla ha minst en klassificerare gemensamt med varandra. En instans av denna klassificerare kan endast delta i en association i taget.



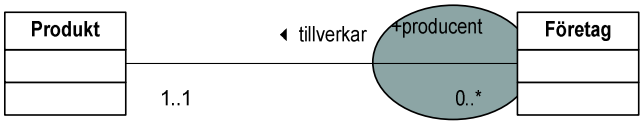
Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

5.1.2 Assocationsnamn

Skrivs ut vid linjen ungefär mitt emellan de sammankopplade klasserna (inte vid någondera änden eftersom associationens namn då kan förväxlas med rollnamnen). Namnsträngen kan även kompletteras med en liten fylld (svart) triangel som visar i vilken riktning namnet skall utläsas. Riktningen har dock ingen semantisk betydelse utan är enbart beskrivande. De klasser som sammankopplas genom associationen ordnas i den riktning som anges i samband med namnet. Ett nyckelord (stereotyp) kan anges i samband med associationens namn (inom guillemet << >>). Associationer behöver ej namnges (framför allt inte om syftet framgår av sammanhanget).

5.1.3 Associationsände

Med associationsände avses den delen av en association som är i direktkontakt med en klassificerare. Varje association har minst två ändar.



Även en associationsände kan ha olika utsmyckningar. Dessa fästs i närheten av änden. Följande varianter finns:

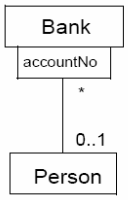
- Multiplicitet (eng. Multiplicity) – Antalet möjliga objekt av en viss klass som kan kopplas samman med ett objekt av en annan klass.

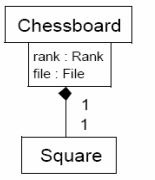
- Ordning (eng. Ordering) – I de fall multipliciteten är större än ett kan de relaterade elementen vara ordnade eller oordnade. Anges ingenting anses elementen vara oordnade. Begräsningar (eng. Constraints) kan användas för att specificera olika sätt att ordna elementen på. Exempel på möjliga begränsningar är: oordnade, ordnade (inga dubletter) och sorterade. Exakt hur ordningen implementeras beror på programspråket.

- Kvalificerare (eng. Qualifier) – Användande av kvalificerare är valfritt men när de används måste de visas i diagrammet. En kvalificerare är

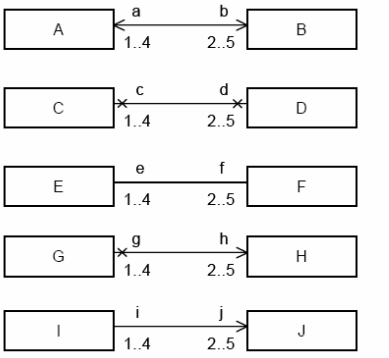
Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

närmast att betrakta som en nyckel i en databastabell, d.v.s. ett eller flera attribut som kan användas för att identifiera objektet. Kvalificerare används (nästan) enbart i relationerna ”en till många” eller ”många till många” mellan klasserna och syftet är då att man genom att använda sig av kvalificeraren skall kunna identifiera ett unikt objekt. Det innebär att den angivna multipliciteten vid en kvalificerare syftar på antal objekt som kan identifieras med hjälp av denna kvalificerare. I exemplet till höger så kan man med hjälp av kontonumret identifiera personen och med hjälp av rad och kolumn identifiera positionen på schackbrädet.





- Navigerbarhet (eng. Navigability) – En pil kan kopplas fast på associationens ände för att visa att navigering gentemot den klassificeraren stöds. Pilar kan sättas fast på noll, en eller två ändar. Det är valfritt att visa pilarna i diagrammet. Navigerbarheten kan vara definierad (visas med pilar i den riktning man kan navigera), eller odefinierad (utan pilar). Vill man visa att man enbart kan navigera i en riktning visas det med hjälp av ett kryss vid den klass som man inte kan navigera till. Kan man navigera i båda riktningar visar man det med pilhuvuden i båda riktningarna.



- Aggregering (eng. Aggregation indicator) – En ofylld diamant används för att visa på aggregering. Diamanten kan endast fästas på en av associationens ändar och fäst vid den klass som representerar aggregatet (helheten). Är diamanten fylld visar den på en starkare variant av aggregering som kallas komposition. Aggregat är valfria att använda sig av men måste visas i diagrammet när de väl används.

- Rollnamn (eng. Rolename) – Rollnamnet beskriver den roll som spelas av den klass som är sammankopplad med en av ändarna. Det är valfritt att använda sig av rollnamn men de måste visas i diagrammet när de väl används.

- Specificering av gränssnitt (eng. Interface specifier) – Namnet på en klassificerare enligt syntaxen ”:klassnamn, …” Gränssnittet visar på det beteende som är en förutsättning för att associationen skall vara möjlig.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

- Förändringsbarhet (eng. Changeability) – Om länkarna mellan objekten går att förändra (lägga till, ta bort och/eller flyttas) behövs ingen indikator. Egenskapen {frozen} visar att inga länkar kan förändras. Detta gäller den associationsände där nyckelordet är placerat. Egenskapen {addOnly} visar att länkar kan läggas till men inte förändras eller tas bort.

- Synlighet (eng. Visibility) – Visas antingen med explicita namn som {public} eller med tecken som +, #, - framför rollnamnet. Synligheten syftar på associationen.

- Övriga egenskaper – Ytterligare egenskaper kan specificeras men det finns ingen fördefinierad grafisk symbol för dessa. Istället används begränsningar i närheten av associationens ände (t.ex. {mutability}).

5.1.4 Multiplicitet

Multipliciteten specificerar den tillåtna kardinaliteten mellan olika objekt och definieras med hjälp av en kommaseparerad sekvens av heltal. En asterisk kan användas för att visa ett obegränsat övre antal. En ensam asterisk (*) är jämställt med 0..* (noll eller fler).

Multiplicitet bör specificeras i ökande ordning (1..3, 7, 10 istället för 7, 10, 1..3) och intervallet bör vara så stort som möjligt (0..2 istället för 0, 1, 2)

Rollnamn och multiplicitet bör placeras nära linjens ände så att de ej riskerar att förväxlas med andra associationer. De kan placeras såväl under som över associationslinjen.

Med kardinalitet avses antalet element i ett visst sammanhang. När multipliciteten är 0..1 kan kardinaliteten vara antingen 1 eller 0. Detta innebär att om multipliciteten är t.ex. 1..3 behöver inte antalet länkar vara samma hela tiden, det räcker om det hela tiden håller sig inom rätt intervall, d.v.s. 1, 2 eller 3.

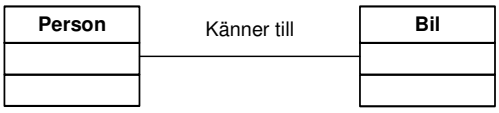
Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

Exempel:

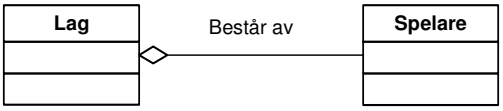
X	Definierar ett fast antal länkar
x..y	Definierar antalet länkar som mellan x och y
x..*	Definierar antalet länkar som mellan x och oändligheten
*	Är lika med 0..*
X, y, z	x, y eller z stycken länkar
X, y..z	x eller mellan y och z stycken länkar

5.1.5 Olika grader av närhet

Utöver association, som är den svagaste kopplingen mellan två objekt, så finns det ytterligare två typer av associationer i UML; kompositioner och aggregat. Dessa används ofta för att visa på delar av en helhet. UML definierar vanliga associationer och kompositioner tämligen exakt men lämnar aggregat (som är mitt emellan de två förstnämnda) lite lösare definierat. Kompositionen är ett specialfall (lite starkare) av ett aggregat. Den vanliga associationen visas som ett streck mellan två eller flera klasser.



5.1.5.1 Aggregat



Enbart binära associtioner kan ingå i ett aggregat. Ett aggregat medför svagt ägarskap. En del kan ingå i flera aggregationer samtidigt och kan även byta ägare över tiden. Vid aggregat är det inte nödvändigt att radera delarna när ägaren raderas. Aggregatet illustreras med en vit diamant på associationsänden vid helheten.

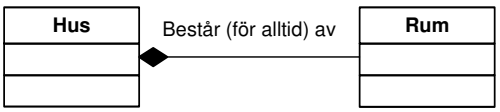
Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

Används t.ex. när objekt fysiskt består av andra objekt, när objekt logiskt kan betraktas som komponenter i en helhet och/eller när objektet tillhör en kategori. Är det tveksamt om objekten är ett aggregat bör man använda ”vanliga” associationer.

Två eller flera aggregeringar till ett och samma aggregat kan visas med hjälp av en trädstruktur genom att aggregeringarna slås samman till en linje i närheten av aggregatet. Detta förutsätter dock att alla associationändens utsmyckningar är identiska och har samma innebörd som om man använder sig av flera linjer.

5.1.5.2 Komposition (eng. Composition)

Komposition är en stark form av aggregat som kräver att ett delobjekt är inkluderat i högst en komposition i taget och att kompositionen har ensamt ansvar för användandet av sina delar. Multipliciteten vid aggregatänden får inte vara större än 1 (delobjekten får ej delas mellan flera olika kompositioner). Varje del av objektmodellen identifieras med en unik komposition. Kompositionen är även transitiv, d.v.s. om objekt A är en del av objekt B som är en del av objekt C så är även objekt A en del av objekt C.



Komposition visas med hjälp av en fylld (svart) diamant vid associationsänden alternativt med grafisk nästling av klassrektanglarna. Vid nästling kan multipliciteten anges i de nästlade klassrektanglarnas övre högra hörn. Utelämnas denna antas den vara ”många”. Det nästlade elementet kan även ha ett rollnamn. Detta anges då enligt ”rollnamn:klassnamn”.

Vid en komposition kan ett delobjekt enbart ingå i en komposition i taget och kompositionen ansvarar ensamt för sina delar (inklusive skapande och borttagande av delobjekten). Tas kompositionen bort måste alla delarna även tas bort, alternativt kan ett delobjekt tilldelas en ny komposition.

Kriterium för komposition: En del kan inte existera utanför kompositionen, d.v.s. instansen av en del är alltid inkluderad i en komposition i taget.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

Ett specialfall är om multipliciteten från delen till kompositionen är 0..1, i så fall kan det finnas delar som existerar utanför kompositionen. I inget annat fall kan någon del av en komposition leva utanför densamma.

Observera att attribut i själva verket är ett slags komposition mellan klassens klassificerare och attributens klassificerare. Dock så skiljer sig ofta syftet och användningen av de två åt. Se tidigare diskussion om skillnaden mellan klasser och attribut.

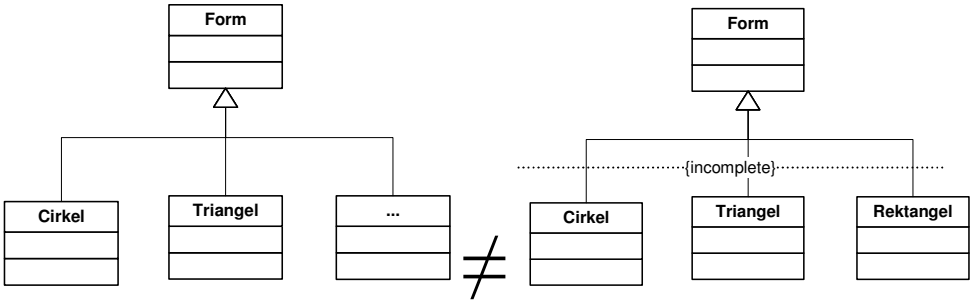
5.2 Att återanvända information med hjälp av generaliseringar (eng. Generalization)

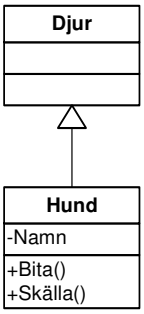
En generalisering är en taxonomisk relation mellan ett mer generellt element (föräldern) och ett mer specifikt element (barnet). Det senare elementet överensstämmer med det förra men lägger till ytterligare information. Begreppet specialisering kan användas istället för generalisering. Ett däggdjur är en generalisering av hund, på samma sätt som hunden är en specialisering av däggdjuret. Valet av benämning generalisering/specialisering beror på i vilken riktning man läser diagrammet. Man kan också säga att den specialiserade klassen (hunden) ärver de gemensamma egenskaperna från den generella klassen (däggdjur). Eftersom alla däggdjur föder levande ungar så gör även hunden det.

Generaliseringar kan definieras för klasser, paket, användningsfall etc. Till och med associationer kan generaliseras även om detta är sällsynt.

Generalisering visas som en heldragen linje från ett barn (det mer specifika elementet, subklassen) till föräldern (det mer generella elementet, superklassen) med en pilhuvud i form av en triangel som fäster vid det mer generella elementet. Ibland väljer man att inte visa alla barnen/subklasserna i ett diagram. Det visas med hjälp av tre punkter (...) istället för klassen. Detta innebär dock inte att nya subklasser kan läggas till senare utan enbart att det redan finns subklasser, men att de ej visas för tillfället.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se





Den generella klassen har de kännetecken som är gemensamma för de specialiserade klasserna. Alla objekt av den specialiserade klassen har de kännetecken som objekt av den generella klassen har.

En generalisering bör kunna utläsas som ”ett slags” eller ”är en”, t.ex. ”en hund är ett djur” eller ”en hund är ett slags djur”. Kan man inte göra detta är det ingen bra generalisering. Observera skillnaden mellan uttrycken ”en hund är ett djur” och ”Karo är en specifik hund”. Det första är en generalisering och det andra en instansiering (dvs Karo är en instans av typen Hund).

Det finns fördefinierade begränsningar som kan appliceras på subklasserna. Detta görs genom en kommaseparerad lista med nyckelord som placeras inom ”måsvingar” antingen vid generaliseringens pilhuvud (triangeln) eller vid en streckad linje som korsar alla olika generaliseringslinjerna. Följande nyckelord kan användas:

- Overlapping – Ett element kan ha två eller flera subklasser ur arvshierarkin som superklasser. En instans kan vara en direkt eller indirekt instans av två eller fler av subklasserna. Hur detta implementeras skiljer från programspråk till programspråk. T.ex. så stödjer inte java och C# multipelt arv.

- Disjoint – Ett element kan **inte** ha två subklasser ur arvshierarkin som superklasser. Ingen instans kan vara en direkt eller indirekt instans av två barn.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

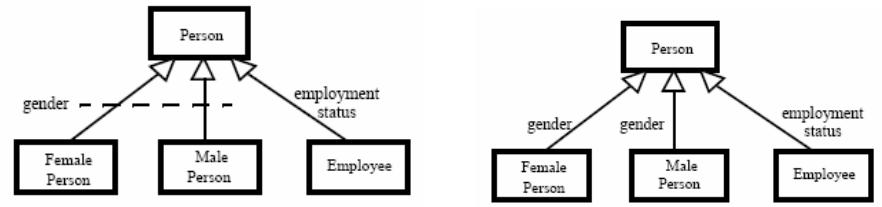
- Complete – Alla subklasser är definierade (oavsett om de visas eller inte). Inga ytterligare specialiseringar väntas.

- Incomplete – Några subklasser är definierade, men det är känt att dessa inte är alla även om inga ytterligare ännu är identifierade/definierade. Det finns ytterligare subklasser som ännu inte är upptagna i modellen. Nyckelordet säger alltså något om själva modellen.

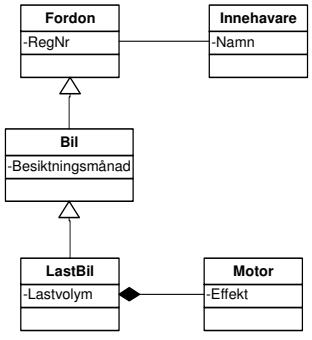
Om inget annat sägs så antas relationen vara ”incomplete” och ”disjoint”. Flera generaliseringar till samma superklass kan slås samman till en trädstruktur med gemensam topp (triangeln). Sätts en etikett på en sådan trädstruktur gäller den för samtliga specialiseringar.

Alla klassens kännetecken ärvs (attribut, operationer, associationer) och ytterligare kännetecken kan läggas till i en specialisering.

En klass kan även specialiseras i olika avseenden (t.ex. så kan en person dels specialiseras utifrån kön, dels utifrån anställningsform. Detta kan visas i diagrammet genom att skriva in namnet på diskriminatorn vid arvsrelationen enligt exemplen nedan.



5.2.1 Exempel på arvsrelationen



Modellen nedan berättar att alla fordon har ett registreringsnummer och eventuellt en innehavare. Innehavaren har ett namn och kan äga noll eller flera fordon (t.ex. en bil och en lastbil).

En bil är ett slags fordon och har därför också ett registreringsnummer och en eventull innehavare. Varje bil har dessutom en motor med en viss effekt.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

En transportbil är en slags bil och har därför såväl registreringsnummer som besiktningsmånad och motor samt en eventuell ägare. Transportbilen har dessutom en angiven lastvolym.

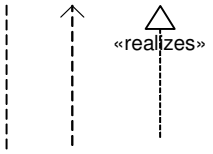
En innehavare kan alltså ha såväl fordon som bilar och lastbilar (eftersom ju dessa är ett slags bilar).

5.3 Beroenden

Ett beroende visar på en semantisk relation mellan två modellelement (eller två uppsättningar modellelement). Relationen innebär att en förändring i det oberoende elementet (målelementet, som pilen pekar mot) kan kräva en förändring i det oberoende elementet (källelementet).

Beroendet visas med hjälp av en streckad pil mellan de två modellelementen. Elementet vid pilens bas (klienten) är beroende av elementet vid pilhuvudet (leverantören). Pilen kan även tilldelas en stereotyp och ett valfritt namn. Det är möjligt att ha flera klienter som är beroende av flera leverantörer. Det visas genom en eller flera pilar som utgår från klienterna och pekar på leverantörerna.

Det är viktigt att skilja på beroenden (streckade pilar), anteckningar/noteringar som visas med hjälp av streckade linjer (utan pilhuvuden) och realiseringar (som visas som streckade arvspilar.



Följande fördefinierade typer av beroenden används i UML's klassdiagram.

- access – Ett paket kan referera till publika element som ägs av ett annat paket.

- bind – Kopplingen mellan en mall klass och den klass som använder sig av mallen. I praktiken innebär det att mallklassens värden kopieras över till klassen som använder sig av mallen.

- trace – Används för att kunna följa ett koncept till olika modeller, t.ex. för att kunna spåra ett krav in i analys- och designmodeller.

Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

- derive – En beräkningsbar relation mellan två element, t.ex. så kan kundreskontrat beräknas utifrån vilka kunder som lagt beställningar..

- import – Ett paket kan referera publika element i ett annat paket samt lägga till de elementens namn till den egna namnrymden.

- refine – En härledning eller förfining mellan två element, t.ex. mellan analys och designklasser. Fungerar ofta som en historisk koppling mellan två element som representerar samma koncept men med olika detaljeringsgrad.

- use – Ett element kräver närvaron av ett annat element för att kunna fungera. T.ex. så kan operationer anropas i det andra elementet.

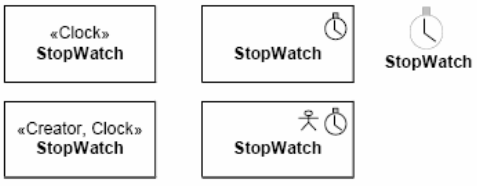
6 Utökningsmekanismer

UML har tre utökningsmekanismer som används för att anpassa UML mer efter en verksamhets specifika behov. Dessa är stereotyper, uppmärkta värden (eng. tagged values) och begränsningar (eng. constraints).

6.1 Stereotyper

En stereotyp är en egendefinierad specificering av ett befintligt UML-element (skall man vara riktigt exakt så är det egentligen en metaklass, men vem behöver sådan precision…). Det kan liknas vid en subklassning av detta. Stereotyper används för att öka precisionen i klassdiagrammet. T.ex. när man vill visa att en klass representerar en databastabell, eller som i exemplet nedan en ”klocka”.

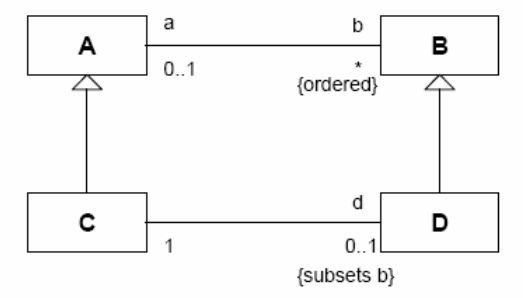
En stereotyp kan visas antingen textuellt (inom guillemet ovanför klassens namn) eller som en ikon i rektangelns övre högra hörn. Objektets stereotyp måste vara i överenstämmelse med klassens stereotyp. Vill man så kan man även helt ersätta den traditionella klassymbolen med en egen ikon.



Systemvaruhuset AB Råsundavägen 166 169 36 Solna
Tel 08 50 52 10 42 Fax 08 50 52 10 10 Org. Nr 556665-7614 E-post info@systemvaruhuset.se www.systemvaruhuset.se

6.2 Begränsningar (constraints)

Begränsningen är den andra av UMLs tre utökningsmekanismer. Den används för att i klassdiagrammet lägga in regler för hur objektdiagrammet kan struktureras. Det finns ett antal fördefinierade begränsningar, plus att användaren själv kan definiera egna. En begränsning presenteras inom måsvingar {}. T.ex. {Subset}. Subset innebär att rollerna i den ena relationen är en delmängd av rollerna i en annan relation. Till exempel så är cheferna en delmängd av de anställda.



6.3 Uppmärkta värden (tagged values)

Stereotyper och begränsningar används för att underlätta beskrivningen av det som modellen försöker att beskriva. Uppmärkta värden däremot används för att lämna information om modellen, snarare än för att underlätta beskrivningen. Stereotypens attribut kallas för uppmärkta värden och dessa kan därför sägas vara metaattribut.

7 Referenser

Den intresserade kan hitta mer information på:

- UML – reference manual
- UML Distilled
- www.uml.org
- www.systemvaruhuset.se/kunskapsbank