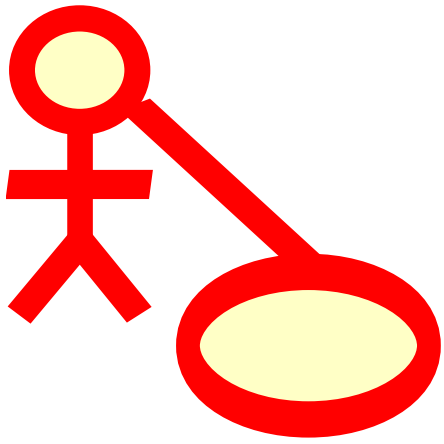
Umbrello UML Modeller-håndbogen



Indhold

Resumé

Umbrello UML Modeller hjælper til med udviklingsprocessen for programmel ved at bruge industristandarden Unified Modelling Language (UML) for at gøre det muligt at oprette diagrammer til at konstruere og dokumentere systemer.

Kapitel 1

Indledning

Umbrello UML Modeller er et UML-diagramværktøj som understøtter dig i udviklingsprocessen af programmel. I særdeleshed under analyse- og konstruktionsfaserne af processen, hjælper Umbrello UML Modeller dig med at oprette et produkt af høj kvalitet. UML kan også bruges til at dokumentere programmelkonstruktioner for at hjælpe dig og dine medudviklere.

At have en god model af programmellet er den bedste måde at kommunikere med andre udviklere som arbejder med projektet og med kunder. En god model er yderst vigtig for middelstore og store projekter, men er også meget nyttig for små. Selv om du arbejder på et lille enmandsprojekt, har du nytte af en god model, eftersom den giver dig et overblik, som hjælper dig til at kode det rigtige fra begyndelsen.

UML er et diagramsprog som bruges til at beskrive sådanne modeller. Du kan repræsentere dine idéer i UML med forskellige slags diagrammer. Umbrello UML Modeller 1.2 understøtter følgende typer:

- Klassediagram
- Sekvensdiagram
- Samarbejdsdiagram
- Brugstilfældediagram
- Tilstandsdiagram
- Aktivitetsdiagram
- Komponentdiagram
- Udplaceringsdiagram

Mere information om UML findes på hjemmesiden OMG, http://www.omg.org, dem der lavede UML-standarden.

Vi håber at du vil nyde at arbejde med Umbrello UML Modeller, og at den hjælper dig med at lave programmel af høj kvalitet. Umbrello UML Modeller er et frit værktøj, og det eneste som vi beder dig om er at rapportere eventuelle fejl, problemer eller forslag til Umbrello UML Modellers udviklere på uml-devel@lists.sourceforge.net eller http://bugs.kde.org.

Kapitel 2

UML, det basale

2.1 Om UML

Dette kapitel giver en hurtig oversigt over det basale i UML. Husk at dette ikke er en fuldstændig UML-vejledning. Hvis du vil lære dig mere om Unified Modelling Language, eller om almen analyse og konstruktion af programmel, henvises du til en af de mange bøger som er tilgængelige om emnet. Der er også mange vejledninger på internettet, som du kan bruge som et startpunkt.

Unified Modelling Language (UML) er et diagrambaseret sprog eller notation til at specificere, visualisere og dokumentere modeller af objektorienteret programmel. UML er ikke en udviklingsmetode, hvilket betyder at den ikke fortæller for dig hvad du skal gøre først og hvad du skal gøre derefter, eller hvordan du skal konstruere dit system, men den hjælper til at visualisere konstruktionen og kommunikere med andre. UML styres af Object Management Group (OMG), og er en industristandard for at beskrive modeller af programmel.

UML er konstrueret for design af objektorienteret programmel, og har begrænset brug for andre programmeringsparadigmer.

UML er opbygget af mange modelleringselementer som repræsenterer forskellige dele af programmelsystemet. UML-elementerne bruges til at oprette diagrammer, som repræsenterer en vis del, eller en synspunkt for systemet. Følgende slags diagrammer understøttes af Umbrello UML Modeller:

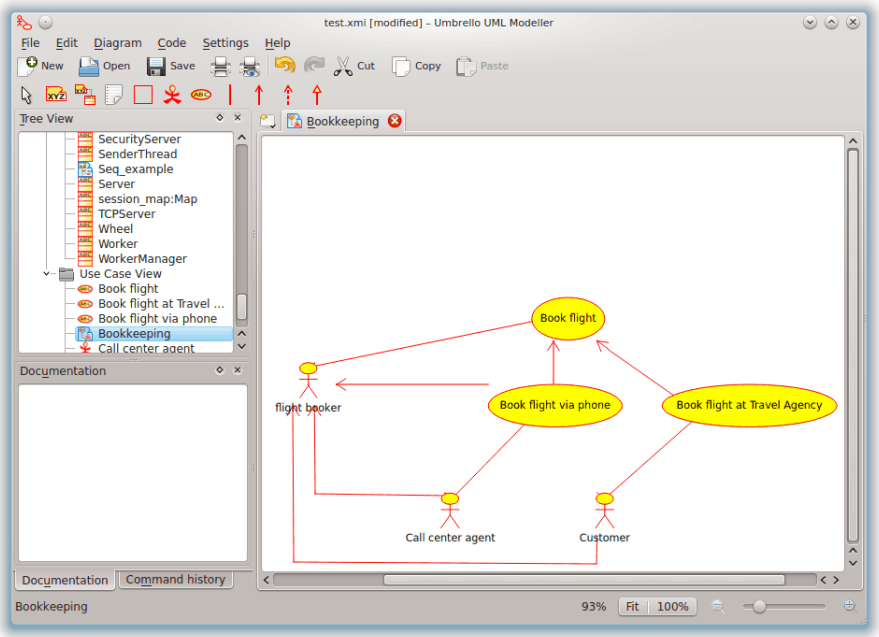
- *Brugstilfælde-diagrammer* viser aktører (mennesker eller andre brugere af systemet), brugstilfælde (scenarier når de bruger systemet), og deres forhold
- *Klassediagrammer* viser klasser, og forholdene mellem dem
- *Sekvensdiagrammer* viser objekter og deres og en sekvens af metodekald de laver til andre objekter.
- *Samarbejdsdiagrammer* viser objekter og deres forhold, med betoning på objekter som deltager i udveksling af meddelelser
- *Tilstandsdiagrammer* viser tilstande, tilstandsændringer og begivenheder for et objekt eller en del af systemet
- *Aktivitetsdiagrammer* viser aktiviteter, tilstander og tilstandsændringer for objekter og begivenheder som sker i en del af systemet
- *Komponentdiagrammer* viser programmeringskomponenter på højt niveau (såsom Kparts eller Java Beans).
- *Udplaceringsdiagrammer* viser komponenternes instanser og deres indbyrdes forhold.

2.2 UML-elementer

2.2.1 Brugstilfældediagram

Brugstilfældediagrammer beskriver forhold og afhængighed mellem en gruppe *brugstilfælde* og aktører som deltager i processen.

Det er vigtigt at observere at brugstilfældesdiagrammer ikke er passende til at repræsentere konstruktioner, og ikke kan beskrive systemets indre dele. Brugstilfældediagrammer er beregnede til at muliggøre kommunikation med fremtidige brugere af systemet, og med kunden. De er til særlig hjælp for at afgøre hvilke funktioner som det kræves at systemet skal have. Med andre ord fortæller brugstilfældediagrammer om *hvad* systemet skal gøre, men de angiver ikke — og kan ikke angive — *hvordan* dette skal opnås.



Umbrello UML Modeller som viser et brugstilfældediagram

2.2.1.1 Brugstilfælde

Et *brugstilfælde* beskriver — fra aktørernes synvinkel — en samling aktiviteter i et system, som giver anledning til et konkret, væsentligt resultat.

Brugstilfælde er beskrivelser af typisk vekselvirken mellem brugerne af et system og systemet selv. De repræsenterer systemets ydre grænseflade, og angiver en slags krav på hvad systemet skal gøre (husk, kun hvad, ikke hvordan).

Ved arbejde med brugstilfælde, er det vigtigt at huske nogle enkle regler:

- Hvert brugstilfælde hører sammen med mindst en aktør
- Hvert brugstilfælde har et ophav (dvs. en aktør)
- Hvert brugstilfælde leder til et relevant resultat (et resultat med 'forretningsværdi').

Brugstilfælde kan også have forhold til andre brugstilfælde. De tre mest typiske slags forhold mellem brugstilfælde er:

- «*include*» (indeholder), hvilket angiver at brugstilfældet finder sted *inde i* et andet brugstilfælde
- «*extends*» (udvider), hvilket angiver at i visse tilfælde, eller i et tilfælde (som kaldes et udvidelsespunkt), bliver et brugstilfælde udvidet af et andet.
- *Generalisering* angiver at et brugstilfælde arver egenskaberne for 'super'-brugstilfældet, og kan sætte nogen af dem ud af kraft, eller tilføje nye på samme måde som arv mellem klasser.

2.2.1.2 Aktør

En aktør er en ekstern enhed (udenfor systemet) som vekselvirker med systemet ved at deltage i (og ofte indlede) et brugstilfælde. Aktører kan i virkeligheden være mennesker (for eksempel brugere af systemet), andre maskinesystemer eller ydre begivenheder.

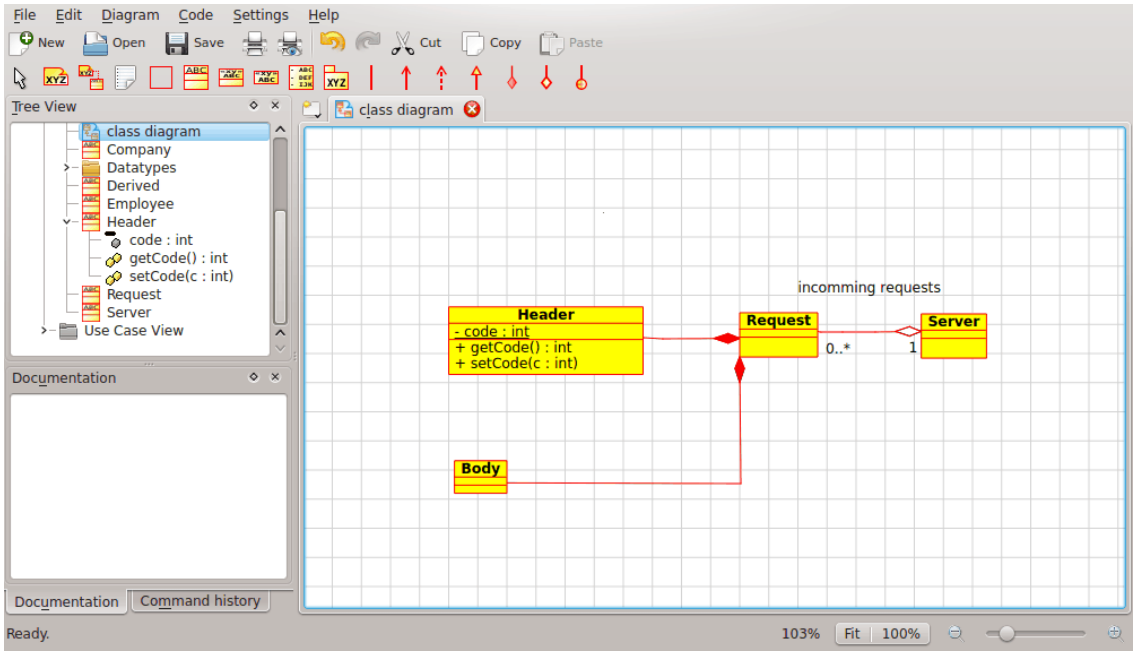
Aktører repræsenterer ikke *fysiske* mennesker eller systemer, men deres *rolle*. Det betyder at når en person vekselvirker med systemet på forskellige måder (antager forskellige roller) repræsenteres han ved flere aktører. En person som for eksempel giver kundeunderstøttelse via telefon og tager imod bestillinger fra kunden til systemet, vil blive repræsenteret af en aktør 'Kundeunderstøttelsespersonale' og af en aktør 'Salgsrepræsentant'.

2.2.1.3 Beskrivelse af brugstilfælde

En beskrivelse af et brugstilfælde er en tekstbaseret beretning om brugstilfældet. Det er ofte i form af en note eller et dokument som på en eller anden måde er linket til brugstilfældet, og forklarer processerne eller aktiviteterne som finder sted i brugstilfældet.

2.2.2 Klassediagram

Klassediagrammer viser de forskellige klasser som opbygger et system og hvordan de relateres til hinanden. Klassediagrammer siges at være 'statiske' diagrammer, fordi de viser klasserne, sammen med deres metoder og attributter, samt det statiske forhold mellem dem: hvilke klasser der 'kender til' andre klasser, eller hvilke klasser der 'er en del' af andre klasser, men viser ikke metodekald mellem dem.

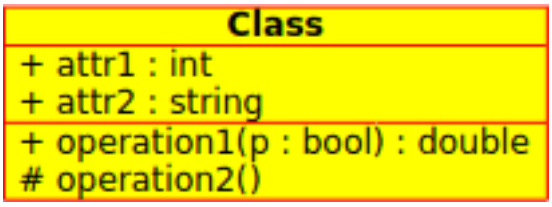


Umbrello UML Modeller som viser et klassediagram

2.2.2.1 Klasse

En klasse definerer attributterne og metoderne for en mængde af objekter. Alle objekter af klassen (instanser af klassen) deler samme opførsel, og har samme mængde af attributter (hvert objekt har sin egen mængde). Udtrykket 'type' bruges ind imellem i stedet for klasse, men det er vigtigt at nævne at de to ikke er det samme, og at type er et mere generelt udtryk.

Klasser i UML repræsenteres af rektangler, med klassens navn, og kan også vise klassens attribut og operationer i to 'afdelinger' inde i rektanglet.



Visuel repræsentation af en klasse i UML

2.2.2.1.1 Attribut

Attributter i UML vises i det mindste med deres navne, og kan også vises med type, oprindelig værdi og andre egenskaber. Attribut kan også vises med synlighed:

- \+ Står for *åbne* (public) attributter
- # Står for *beskyttede* (protected) attributter
- \- Står for *private* (private) attributter

2.2.2.1.2 Operationer

Operationer (metoder) vises også i det mindste med deres navne, og kan også vises med parametre og returtyper. Operationer, præcis som attributter, kan vises med deres synlighed:

- \+ Står for *åbne* (public) operationer
- # Står for *beskyttede* (protected) operationer
- \- Står for *private* (private) operationer

2.2.2.1.3 Skabeloner

Klasser kan have skabeloner, en værdi som bruges for en uspecificeret klasse eller type. Skabelontypen angives når klassen initieres (dvs. et objekt laves). Skabeloner findes i moderne C++ og vil blive introduceret i Java 1.5, hvor de kaldes Generics.

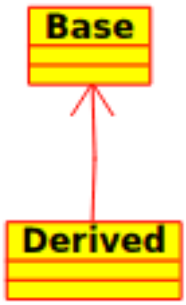
2.2.2.2 Klasseassociationer

Klasser kan relateres til (associeres med) hinanden på forskellige måder:

2.2.2.2.1 Generalisering

Arv er et af de grundlæggende begreber i objektorienteret programmering, hvor en klasse 'opnår' alle attributter og operationer fra klassen den arver fra, og kan overskride/ændre nogen af dem, samt tilføje flere egne attributter og operationer.

En *generalisering* mellem to klasser i UML, placerer dem i et hierarki som repræsenterer arvebegrebet for en afledt klasse fra en basisklasse. Generaliseringer i UML repræsenteres med en linje som binder de to klasser sammen, med en pil på basisklassens side.



Visuel repræsentation af en generalisering i UML

2.2.2.2.2 Associationer

En association repræsenterer et forhold mellem klasser, og giver den fælles semantik og struktur for mange typer af 'forbindelser' mellem objekter.

Associationer er mekanismen som tillader at objekter kommunikerer med hinanden. De beskriver forbindelsen mellem forskellige klasser (forbindelsen mellem de virkelige objekter kaldes objektforbindelser, eller *link*).

Associationer kan have en rolle, som angiver associationens formål, og kan være ensrettede eller gensidige (indikerer om de to objekter som deltager i forholdet kan sende meddelelser til hinanden, eller om kun et af dem kender til det andet). Hver eneste af associationerne har også en multiplicitet, som bestemmer hvor mange objekter på denne side af associationen kan relateres til et objekt på den anden side.

Associationer i UML repræsenteres som linjer som binder de klasser sammen som deltager i forholdet, og kan også vise rollen og multipliciteten for hver af deltagerne. Multiplicitet vises som et interval [minimum..maksimum] med ikke-negative værdier, med en stjerne (*) på maksimumsiden som repræsenterer uendeligt.

Visuel repræsentation af en association i UML

2.2.2.2.3 Aggregering

Aggregeringer er en særlig slags association, hvor de to deltagende klasser ikke har en ligeværdig status, men udgør et 'helhed-del' forhold. En aggregering beskriver hvordan klassen som intager rollen som helhed, er sammensat af (har) andre klasser, som intager rollerne som dele. Klassen der virker som helhed har altid multiplicitet en, for aggregeringer.

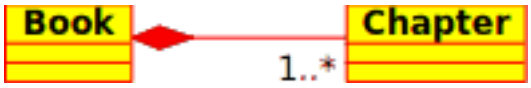
Aggregeringer i UML repræsenteres ved en association som viser en rombe på den side som hører til helheden.

Visuel repræsentation af en aggregeringsrelation i UML

2.2.2.2.4 Sammensætning

Sammensætninger er associationer som repræsenterer *meget stærke* aggregeringer. Det betyder at sammensætninger også former helhed-del forhold, men at forholdet er så stærkt at delene ikke kan eksistere for sig selv. De findes kun inde i helheden, og hvis helheden forstyrres, forsvinder delene også.

Sammensætninger i UML repræsenteres af en udfyldt rombe på siden af helheden.



2.2.2.3 Andre punkter i klassediagrammer

Klassediagrammer kan indeholde flere andre objekter foruden klasser.

2.2.2.3.1 Grænseflader

Grænseflade er abstrakte klasser hvilket betyder at instanser ikke direkte kan skabes fra dem. De kan indehold operationer men ingen attributter. Klasser kan arve fra grænseflader (via en realisationsassociation) og instanser kan derefter laves af disse diagrammer.

2.2.2.3.2 Datatyper

Datatyper er primitiver som typisk er indbyggede i et programsprog. Almindelige eksempler omfatter heltal og en boolesk type. De kan ikke have forhold til klasser, men klasser kan have forhold til dem.

2.2.2.3.3 Gentagelsestyper

Gentagelsestyper er enkle lister med værdier. Et typisk eksempel er en nummereringstype af ugedage. Tilvalgene for en gentagelsestype kaldes Enum Literals. Som datatyper kan de ikke have forhold til klasser, men klasser kan have forhold til dem.

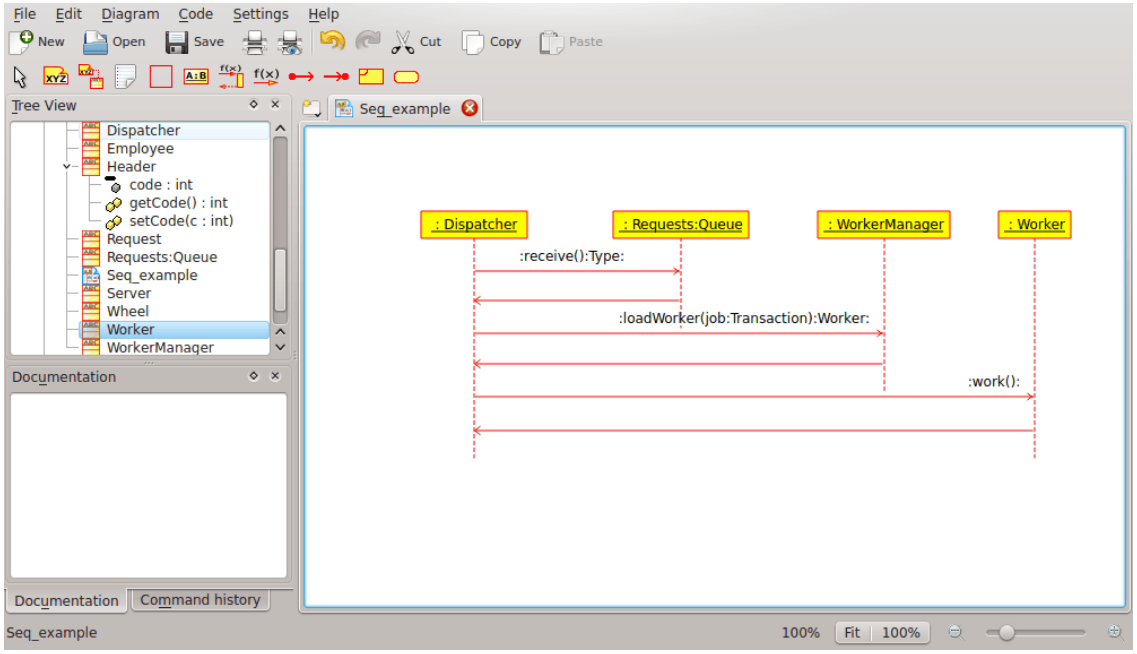
2.2.2.3.4 Pakker

Pakker repræsenterer navnerum i et programsprog. I et diagram bruges de til at repræsentere dele af et system som indeholder mere end en klasse, måske hundredvis af klasser.

2.2.3 Sekvensdiagrammer

Sekvensdiagrammer viser udveksling af meddelelser (dvs. metodekald) mellem flere objekter, i en specifik, tidsbegrænset situation. Sekvensdiagrammer lægger særlig vægt på rækkefølgen og tiden når meddelelserne til objekter sendes.

Objekter repræsenteres af lodrette stregede linjer i sekvensdiagrammer, med objektets navn øverst. Tidsakslen er også lodret, og vokser nedad, så meddelelser sendes fra et objekt til et andet i form af pile med operationer og parameternavn.



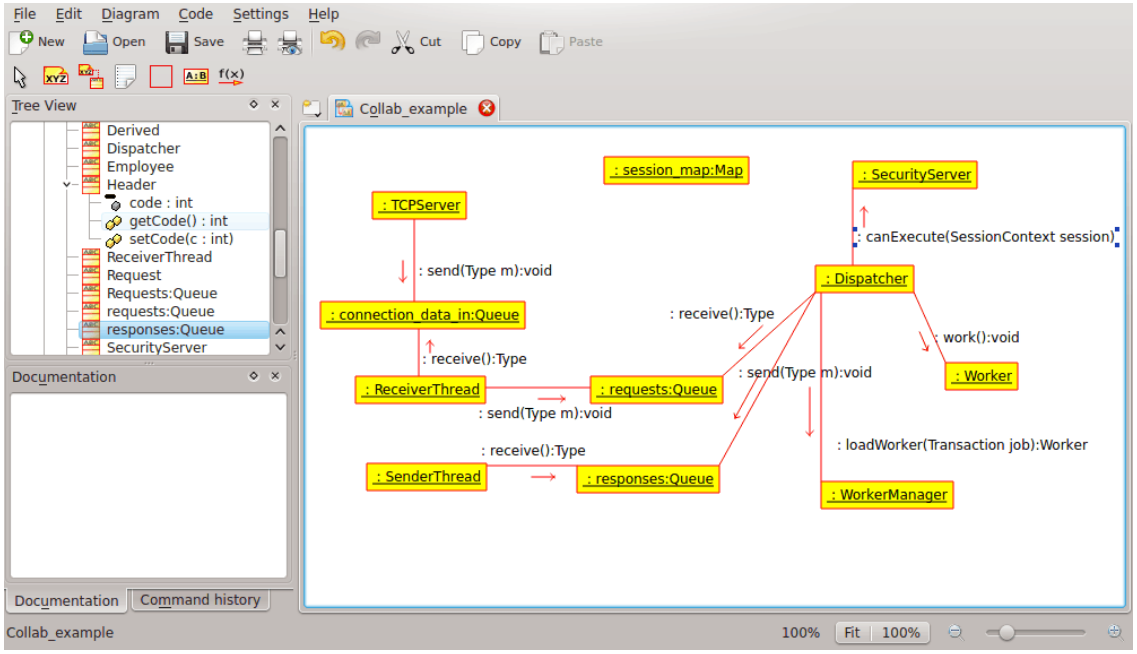
Umbrello UML Modeller som viser et sekvensdiagram

Meddelelser kan enten være synkrone, den normale type for meddelelseskald hvor kontrollen overgår til det kaldte objekt til metoden er kørt færdigt, eller asynkrone hvor kontrollen går direkte tilbage til det kaldende objekt. Synkrone meddelelser har et lodret felt ved siden af det kaldte objektet, for at vise programkontrollen.

2.2.4 Samarbejdsdiagrammer

Samarbejdsdiagrammer viser vekselvirkningen mellem objekter som deltager i en speciel situation. Dette er mere eller mindre samme information som vises i sekvensdiagrammer, men hvor vægten lægges på hvordan vekselvirkningen sker i tiden, mens samarbejdsdiagrammer lægger vægten på forholdet mellem objekterne og deres topologi.

I samarbejdsdiagrammer repræsenteres meddelelser fra et objekt til et andet med pile, som viser meddelelsens navn, parametre og meddelelsesekvensen. Samarbejdsdiagrammer er særligt passende til at vise en særlig programflydning eller situation, og er blandt de bedste diagramtyper til hurtigt at demonstrere eller forklare en process i programmets logik.



Umbrello UML Modeller som viser et samarbejdsdiagram

2.2.5 Tilstandsdiagram

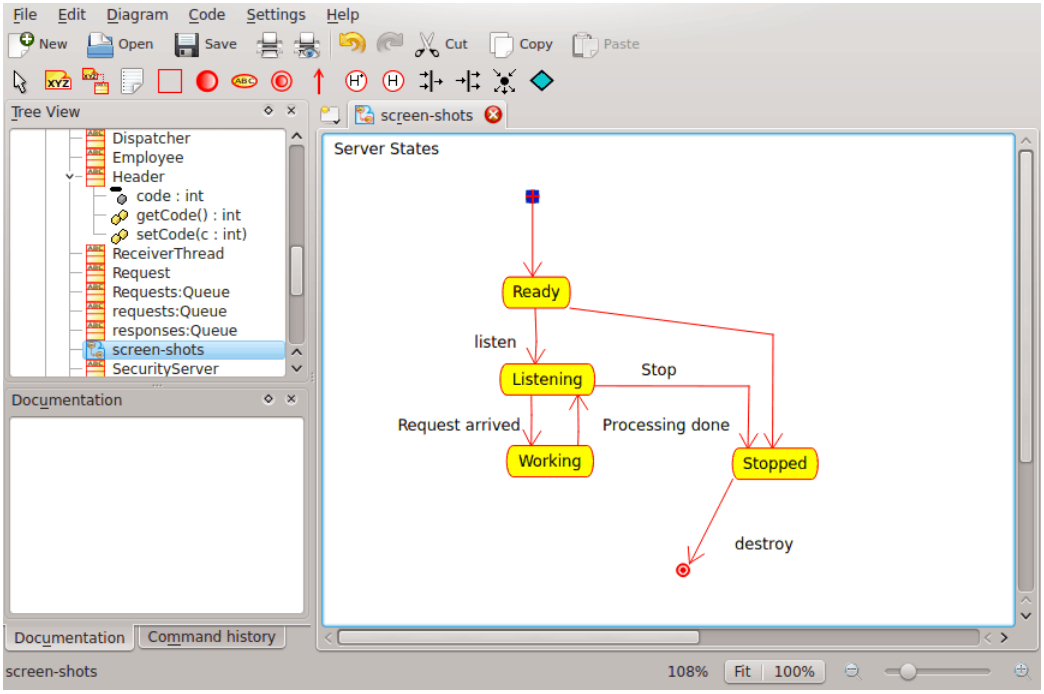
Tilstandsdiagrammer viser de forskellige tilstande et objekt har i sin livstid, og de stimuli som forårsager at objektet ændrer sin tilstand.

Tilstandsdiagrammer ser objekter som *tilstandsmaskiner* eller finite automates, som kan være i en af en mængde begrænsede tilstande og som kan ændre tilstand via et af et begrænset antal stimuli. Et objekt af typen *Netserver*, kan for eksempel være i en af følgende tilstande i sin livstid:

- Klar
- Lytter
- Arbejder
- Stoppet

og begivenhederne som kan gøre at et objekt skifter tilstand er

- Objektet laves
- Objektet tager imod meddelelsen at lytte
- En klient beder om en forbindelse via netværket
- En klient afslutter en forespørgsel
- En forespørgsel køres og afsluttes
- Objektet tager imod meddelelsen at stoppe
- osv



Umbrello UML Modeller som viser et tilstandsdiagram

2.2.5.1 Tilstand

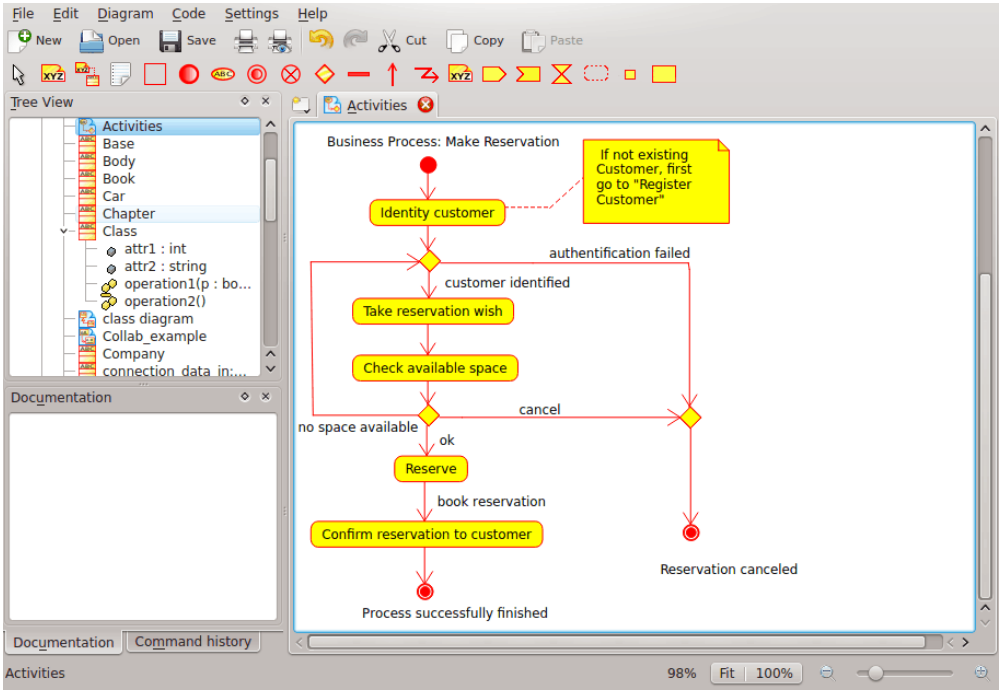
Tilstand er byggeblokken i tilstandsdiagrammer. En tilstand hører til nøjagtig en klasse, og repræsenterer en sammenfatning af de værdier klassens attributter kan intage. En UML-tilstand beskriver den interne tilstand for et objekt af en vis klasse.

Bemærk at ikke hver ændring af en af et objekts attributter skal repræsenteres som en tilstand, men kun de ændringer som væsentligt kan påvirke objektets arbejde.

Der er to specielle typer tilstand: start og slut. De er specielle på den måde at der ikke er nogen begivenhed som kan gøre at et objekt går tilbage til sin starttilstand, og på samme måde er der ingen begivenhed som gør det muligt for et objekt at forlade sin sluttilstand når den først er nået.

2.2.6 Aktivitetsdiagram

Aktivitetsdiagrammer beskriver en følge af begivenheder i et system, ved hjælp af aktiviteter. Aktivitetsdiagrammer er en speciel form af tilstandsdiagrammer, som kun (eller hovedsagelig) indeholder aktiviteter.



Umbrello UML Modeller som viser et aktivitetsdiagram

Aktivitetsdiagrammer ligner procedurelle flydediagrammer, med forskellen at alle aktiviteter er klart linkede til objekter.

Aktivitetsdiagrammer hører altid sammen med en *klasse*, en *operation* eller et *brugstilfælde*.

Aktivitetsdiagrammer understøtter sekventielle- og parallelle aktiviteter. Parallel kørsel repræsenteres med ikonen Del op/Vent, og det er ikke vigtigt for aktiviteter som kører parallelt i hvilken rækkefølge de udføres (de kan køres samtidigt eller en af gangen).

2.2.6.1 Aktivitet

En aktivitet er et enkelt skridt i en process. En aktivitet er en tilstand i systemet med intern aktivitet og i det mindste en udgående overgang. Aktiviteter kan også have mere end en udgående overgang, hvis de har forskellige betingelser.

Aktiviteter kan opbygge hierarkier, hvilket betyder at en aktivitet kan bestå af flere 'detaljeaktiviteter', hvor indkommende og udgående overgange skal passe sammen med de indkommende og udgående overgange i detaljediagrammet.

2.2.7 Hjælpeelementer

Der er nogle få elementer i UML som ikke har noget virkelig semantisk værdi for modellen, men som hjælper til at klargøre dele af diagrammerne. Disse elementer er

- Tekstlinjer
- Noter og ankre
- Bokse

Tekstlinjer er nyttige til at tilføje kort tekstinformation i et diagram. Det er fritstående tekst, og har ingen betydning i selve modellen.

Noter er nyttige til at tilføje mere detaljeret information om et objekt eller en særlig situation. De har den store fordel at noter kan ankres ved UML-elementer for at vise at noten 'hører til' et særligt objekt eller en særlig situation.

Bokse er fritstående rektangler som kan bruges til at gruppere objekter sammen, for at gøre diagrammer mere læsbare. De har ingen logisk mening i modellen.

2.2.8 Komponentdiagrammer

Komponentdiagrammer viser programkomponenter (enten komponentteknologier såsom Kparts, CORBA-komponenter eller Java Beans eller kun dele af systemet som er klart udskillelige) og artefakterne de består af, såsom kildekodefiler, programbiblioteker eller relationsdatabasetabeller.

Komponenter kan have grænseflader (dvs. abstrakte klasser med operationer) som tillader associationer mellem komponenter.

2.2.9 Udplaceringsdiagrammer

Udplaceringsdiagrammer viser komponentinstanserne ved kørsel og deres associationer. De omfatter knuder, som er fysiske ressourcer, typisk en enkelt maskine. De viser også grænseflader og objekter (klasseinstanser).

Kapitel 3

Arbejde med Umbrello UML Modeller

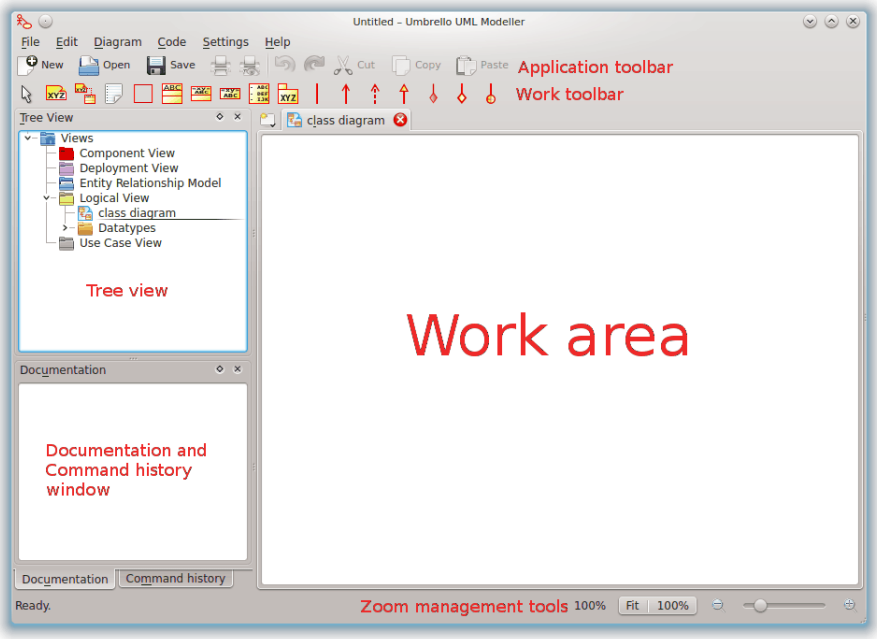
Dette kapitel giver en introduktion til Umbrello UML Modellers brugergrænseflade og fortæller dig alt du skal vide for at begynde med modellering. Alle handlinger i Umbrello UML Modeller er tilgængelige via menuer og værktøjslinjer, men Umbrello UML Modeller bruger også i stor udstrækning sammenhængsafhængige menuer som vises med højre museknap. Du kan højreklikke på næsten alle steder i Umbrello UML Modellers arbejdsområde eller i trævisningen for at få en menu med de mest nyttige funktioner som kan tilpasses til netop det særlige punkt som du arbejder med. Visse brugere synes at dette er lidt forvirrende i begyndelsen (fordi de er mere vant til at arbejde med menuer eller værktøjslinjer), men når man vænnet sig at højreklikke, gør det dit arbejde en hel del hurtigere.

3.1 Brugergrænseflade

Umbrello UML Modellers hovedvindue er opdelt i tre områder som hjælper til at få et overblik over hele systemet og at komme til de forskellige diagrammer hurtigt, under arbejdet med modellen.

Disse områder kaldes:

- Trævisning
- Arbejdsvisning
- Dokumentationsvindue



Umbrello UML Modellers brugergrænseflade

3.1.1 Trævisning

Trævisningen er sædvanligvis placeret længst op til venstre i vinduet, og viser alle diagrammer, klasser, aktører og brugstilfælde som opbygger modellen. Trævisningen lader dig få et hurtig overblik over elementerne som modellen består af. Trævisningen giver også en hurtig måde at skifte mellem de forskellige diagrammer i modellen, og at indsætte elementer fra modellen i det nuværende diagram.

Hvis du arbejder med en model som har mere end et fåtal klasser og diagrammer, kan trævisningen hjælpe dig med at klare det hele ved at organisere modellen i mapper. Du kan oprette mapper ved at vælge passende punkter i den sammenhængsafhængige menu (højreklik på en af mapperne i trævisningen) og du kan organisere elementer ved at flytte dem til passende mapper (træk og slip).

3.1.2 Dokumentationsvindue

Dokumentationsvinduet er det lille vindue placeret længst nede til venstre i Umbrello UML Modeller, som giver en hurtig forhåndsvisning af dokumentationen for objektet som for øjeblikket er markeret. Dokumentationsvinduet er ganske lille, eftersom det er beregnet til at give et hurtigt uddrag fra elementets dokumentation, mens det optager så lidt plads som muligt på skærmen. Hvis du behøver at kigge på dokumentationen i mere detalje, kan du altid åbne punktets egenskaber.

3.1.3 Arbejdsvisning

Arbejdsvisningen er hovedvinduet i Umbrello UML Modeller, og er stedet hvor alle virkelige handlinger sker. Man bruger arbejdsvisningen til at redigere og vise diagrammer i en model. Arbejdsvisningen viser diagrammet som for øjeblikket er aktivt. For øjeblikket kan kun et diagram af gangen vises i arbejdsvisningen.

3.2 Opret, indlæs og gem modeller

Det første du behøver at gøre, for at begynde at gøre noget nyttigt med Umbrello UML Modeller, er at oprette en model at arbejde med. Når du starter Umbrello UML Modeller indlæser den altid den senest brugte model, eller laver en ny, tom, model (afhængig af hvordan du indstiller i indstillingsdialogen). Det gør det muligt at begynde at arbejde direkte.

3.2.1 Ny model

Hvis du på noget tidspunkt behøver at oprette en ny mode, kan du gøre det ved at vælge punktet **Ny** i menuen **Fil**, eller ved at klikke på ikonen **Ny** i programværktøjslinjen. Hvis du for øjeblikket arbejder med en model som er ændret, spørger Umbrello UML Modeller om dine ændringer skal gemmes, inden den nye modellen indlæses.

3.2.2 Gem model

Du kan gemme modellen når som helst, ved at vælge punktet **Gem** i menuen **Fil**, eller ved at klikke på knappen **Gem** i programværktøjslinjen. Hvis du behøver at gemme modellen under et andet navn, kan du bruge punktet **Gem som** i menuen **Fil**.

Af bekvemmelighedsgrunde, tilbyder Umbrello UML Modeller også muligheden for at gemme arbejdet automatisk efter en vis tidsperiode. Du kan indstille om du vil aktivere denne funktion, samt tidsintervallet, i Umbrello UML Modellers **Opsætning**.

3.2.3 Indlæs model

Du kan vælge punktet **Åbn** i menuen **Fil** for at indlæse en eksisterende model, eller klikke på ikonen **Åbn** i programværktøjslinjen. De senest brugte modeller er også tilgængelige i undermenuen **Åbn nyeste** i menuen **Fil**, for at gøre adgangen til de oftest bruge modeller hurtigere.

Umbrello UML Modeller kan kun arbejde med en model af gangen, så hvis du beder programmet indlæse en model for dig, og den nuværende model er ændret siden du sidst gemte den, spørger Umbrello UML Modeller om ændringerne skal gemmes for at forhindre at arbejdet går tabt. Du kan starte to eller flere udgaver af Umbrello UML Modeller på et vilkårligt tidspunkt, du kan også kopiere og indsætte mellem udgaver.

3.3 Redigér modeller

I Umbrello UML Modeller findes der to grundlæggende måder at redigere elementer i modellen.

- Redigér modelelement direkte via trævisningen
- Redigér modelelement direkte via et diagram

Ved brug af den sammenhængsafhængige menu i trævisningen, kan du tilføje, fjerne, og ændre næsten alle elementer i modellen. Højreklik på mapperne i trævisningen for at vise valgmulighederne for at oprette forskellige slags diagrammer, samt - afhængig af om mappen er en *Brugstilfældevisning* eller en *Logisk visning* - aktører, brugstilfælde, klasser osv.

Når du har tilføjet elementer til modellen, kan du også redigere dem ved brug af deres egenskabsdialoger, som du finder ved at vælge punktet *Egenskaber* i den sammenhængsafhængige menu som vises ved et *højre*klik på elementerne i trævisningen.

Du kan også redigere modellen ved at oprette eller ændre elementer via diagrammer. Mere information om hvordan dette gøres, får du i følgende afsnit.

3.4 Tilføj og fjern diagram

UML-modellen består af et sæt UML-elementer og sammenhænge mellem dem. Man kan dog ikke se modellen direkte, men man bruger *Diagrammer* til at kigge på den.

3.4.1 Opret diagrammer

For at oprette et nyt diagram i modellen, vælges helt enkelt diagramtypen du behøver i undermenuen **Ny** fra menuen **Diagram**, og den gives et navn. Diagrammet oprettes, og gøres aktivt, og du ser det med det samme i trævisningen.

Husk at Umbrello UML Modeller i stor udstrækning bruger sammenhængsafhængige menuer: du kan også højreklikke på en mappe i trævisningen, og vælge en passende diagramtype i undermenuen **Ny** fra den sammenhængsafhængige menu. Bemærk at du kun kan oprette brugstilfældediagrammer i brugstilfældemapper, og at de øvrige typer diagrammer kun kan oprettes i mapper for logiske visninger.

3.4.2 Fjern diagram

Skulle du behøve at fjerne et diagram fra modellen, kan du gøre det ved at gøre det aktivt og vælge **Fjern** i menuen **Diagram**. Du kan også opnå dette ved at vælge **Slet** i den sammenhængsafhængige menu for diagrammet i trævisningen.

Eftersom at fjerne et diagram er noget alvorligt, som kunne forårsage at arbejde går tabt, hvis det gøres ved en fejl, beder Umbrello UML Modeller om at du bekræfter en sletningshandling inden diagrammet virkelig fjernes. Så snart et diagram er taget bort, og filen er gemt, er der ingen måde at fortryde handlingen.

3.4.3 Omdøbning af diagrammer

Hvis du vil skifte navn på et eksisterende diagram, kan du let gøre det ved at vælge punktet Omdøb i den sammenhængsafhængige menu i trævisningen.

En anden måde at omdøbe et diagram er via dets egenskabsdialog, som du opnår ved at vælge Egenskaber fra den sammenhængsafhængige menu, eller ved at dobbeltklikke på det i trævisningen.

3.5 Redigér diagram

Mens du arbejder med et diagram, forsøger Umbrello UML Modeller at lede dig rette ved at bruge nogle enkle regler om hvilke elementer er gyldige i forskellige slags diagrammer, samt hvilke forhold som kan eksistere mellem dem. Hvis du er ekspert på UML, kommer du formodentlig ikke til endog bemærke dette, men det er til hjælp for nybegyndere for at oprette diagram som følger standarden.

Så snart du har oprettet diagrammerne er det tiden at begynde redigere dem. Bemærk her (den for nybegyndere subtile) forskel mellem at redigere et diagram, og at redigere *modellen*. Som du allerede ved til, er diagrammer *visninger* af modellen. Hvis du for eksempel laver en klasse ved at redigere et klassediagram, redigerer du i virkeligheden både diagrammet og modellen. Hvis du ændrer farve eller andre visningspunkter for en klasse i klassediagrammet, redigerer du kun diagrammet, men ingenting ændres i modellen.

3.5.1 Indsæt elementer

En af de første ting du gør når du redigerer et nyt diagram, er at indsætte elementer i det (klasser, aktører, brugstilfælde, osv.). Der er to grundlæggende måder at gøre dette:

- Træk eksisterede elementer til modellen fra trævisningen
- Opret nye elementer i modellen, og tilføj dem til diagrammet samtidigt, ved at bruge et af redigeringsværktøjerne i arbejdsværktøjslinjen.

For at indsætte elementer som allerede findes i modellen, trækkes de blot fra trævisningen og slippes der hvor du ønsker at de skal være i diagrammet. Du kan altid flytte element rundt i diagrammet med markeringsværktøjet.

Den anden måde at tilføje elementer til diagrammet er at bruge arbejdsværktøjslinjens redigeringsværktøj (observér at dette også tilføjer elementerne til modellen).

Arbejdsværktøjslinjen var normalt placeret længst til højre for programvinduet, men Umbrello UML Modeller 1.2 har flyttet den længst op i vinduet. Du kan dokke den på den anden side, eller lade den flyde omkring hvis du foretrækker det. Værktøjerne som er tilgængelige på denne værktøjslinjen (knapperne du ser på den) ændres afhængig af hvilket diagram du arbejder med for øjeblikket. Knappen for værktøjet som er valgt lige nu er aktiveret i værktøjslinjen. Du kan skifte til **markeringsværktøjet** ved at trykke på **Esc**-tasten.

Når du har valgt et redigeringsværktøj i arbejdsværktøjslinjen (for eksempel værktøjet til at indsætte klasser), ændres musemarkøren til et kryds, og du kan indsætte elementer i modellen ved at enkeltklikke i diagrammet. Bemærk at element i UML skal have et *entydigt navn*. Så hvis du har en klasse i et diagram som hedder 'KlassA', og senere bruger værktøjet til at indsætte en klasse i et andet diagram, kan du ikke også give den nye klassen navnet 'KlassA'. Hvis det er meningen at de to skal være forskellige elementer, skal du give dem entydige navne. Hvis du forsøger at tilføje *samme* element til diagrammet, er værktøjet for at indsætte klasser ikke det rette værktøj til dette. Du skal i stedet trække og slippe klassen fra trævisningen.

3.5.2 Slet elementer

Du kan slette et hvilket som helst element, ved at vælge punktet **Slet** i dets sammenhængsafhængige menu.

Igen det er *stor* forskel mellem at fjerne et objekt fra diagrammet, og at fjerne et objekt fra modellen. Hvis du fjerner et objekt indefra et diagram, tager du det kun væk fra dette diagram: elementet er stadigvæk en del af modellen og hvis der er andre diagrammer som bruger samme element, vil de ikke lide nogen ændring. På den anden side, hvis du fjerner elementet i trævisningen, tager du i virkeligheden elementet væk fra *modellen*. Eftersom elementet ikke længere eksisterer i modellen, tages det også automatisk væk fra alle diagrammer det vises i.

3.5.3 Redigér elementer

Du kan redigere de fleste UML-elementer i model og diagrammer ved at åbne dets egenskabsdialog og vælge passende punkter. For at redigere egenskaberne for et objekt, vælges **Egenskaber** i dets sammenhængsafhængige menu (højreklik). Hvert element har en dialog som består af flere sider hvor du kan indstille valgmulighederne som har med dette element at gøre. For visse elementer, såsom aktører, kan du kun angive et fåtal muligheder, såsom objektnavn og dokumentation, mens for andre elementer, såsom klasser, kan du redigere dets attribut og operationer, vælge hvad du vil vise i diagrammet (hel operationsunderskriften eller kun operationsnavn, osv.) og til og med farverne du vil bruge for linjer og udfyldningen af klassens repræsentation i et diagram.

For de fleste UML-elementer kan du også åbne egenskabsdialogen ved at dobbeltklikke på det, hvis du bruger markeringsværktøjet (pilen). En undtagelse fra dette er associationer, hvor et dobbeltklik laver et ankerpunkt. For associationer skal du bruge den sammenhængsafhængige menu som vises med højreklik, til at få egenskabsdialogen frem.

Bemærk at du også kan vælge punktet egenskaber i den sammenhængsafhængige menu for elementer i trævisningen. Dette lader dig også redigere egenskaber for diagrammer, såsom at indstille om gitteret skal vises eller ej.

3.5.4 Redigér klasser

Selv om redigering af egenskaber for alle objekter allerede er afdækket i foregående afsnit, fortjener klasser et særligt afsnit, eftersom de er noget mere komplicerede, og har flere valgmuligheder end de fleste andre UML-elementer.

I klassens egenskabsdialog kan du indstille alting, fra farven den bruger til operationerne og attributten den har.

3.5.4.1 Almene klasseindstillinger

Siden med generelle klasseindstillinger i egenskabsdialogen er selvforklarende. Her kan du ændre klassens navn, synlighed, dokumentation, osv. Denne side er altid tilgængelig.

3.5.4.2 Indstillinger af klasseattribut

På siden for indstillinger af attribut, kan du tilføje, redigere eller fjerne attributter (variabler) for klassen. Du kan flytte attributter op og ned i listen ved at trykke på piletasterne langs kanten. Denne side er altid tilgængelig.

3.5.4.3 Indstillinger af klasseoperationer

På lignende måde som for indstillinger af klasseattribut, kan du tilføje, redigere eller fjerne operationer for klassen på siden for indstillinger af klasseoperationer. Når du tilføjer eller redigerer en klasseoperation, indskriver du de grundlæggende data i dialogen *Operationsegenskaber*. Hvis du vil tilføje parametre til operationerne, skal du klikke på knappen **Ny parameter**, som viser dialogen *Parameteregenskaber*. Denne side er altid tilgængelig.

3.5.4.4 Klasseskabelonsindstillinger

Denne side lader dig tilføje klasseskabeloner som er uspecificerede klasser eller datatyper. I Java 1.5 kommer de til at hedde Generics.

3.5.4.5 Siden for klasseassociationer

Siden **Klasseassociationer** viser alle klassens associationer i det nuværende diagram. Et dobbeltklik på en association viser dens egenskaber, og afhængig af type af association, kan du ændre visse parametre her, såsom at indstille mangfoldighed og rollenavn. Hvis associationerne ikke tillader at sådanne punkter ændres, er dialogen for associationsegenskaber kun læsbar, og du kan kun ændre dokumentationen som hører sammen med associationen.

Denne side er kun tilgængelig hvis du åbner klasseegenskaberne inde i et diagram. Hvis du vælger klasseegenskaber fra den sammenhængsafhængige menu i trævisningen, er denne side ikke tilgængelig.

3.5.4.6 Siden for klassevisning

På siden **Visningstilvalg**, kan du indstille hvad der skal vises i diagrammet. En klasse kan vises som kun en rektangel med klassenavnet i (nyttigt hvis du har mange klasser i diagrammet, eller for øjeblikket ikke er interesseret i detaljerne for hver klasse), eller så fuldstændige at pakke, stereotyper, attributter og operationer vises med fuldstændig underskrift og synlighed.

Afhængig af mængden af information som du vil se, kan du vælge tilsvarende tilvalg på denne side. Ændringarne du laver her gælder kun *visningsvalgmulighederne* for diagrammet. Det betyder at 'skjulen' af klassens operationer kun gør at de ikke vises i diagrammet, men operationerne er fortsat der som en del af modellen. Dette er kun tilgængeligt hvis du vælger klasseegenskaberne inde i et diagram. Hvis du åbner klasseegenskaberne fra trævisningen, mangler denne side, eftersom sådanne visningsegenskaber ikke giver mening i dette tilfælde.

3.5.4.7 Siden for klassefarver

På siden **Komponentfarve** kan du indstille farverne som du vil have for linjer og udfyldning af kontrollen. Dette giver naturligvis kun mening for klasser som vises i diagrammer, og mangler hvis du åbner klassens egenskabsdialog i trævisningen.

3.5.5 Associationer

Associationer relaterer to UML-objekter til hinanden. Normalt defineres associationer mellem to klasser, men visse typer af associationer kan også findes mellem brugstilfælde og aktører.

For at oprette en association, vælges passende værktøj i arbejdsværktøjslinjen (generel association, generalisering, aggregering, osv.), og enkeltklik på det første element som indgår i associationen. Enkeltklik derefter på det andet element som indgår. Bemærk at dette er to klik, et på hvert af elementerne som indgår i associationen. Det er *ikke* et træk fra et objekt til et andet.

Hvis du forsøger bruge associationer på en måde som ikke tillades af UML-specifikationen, nægter Umbrello at oprette associationen og du får en fejlmeddelelse. Dette vil indtræffe, hvis for eksempel en generalisering findes fra klasse A til klasse B, og du derefter forsøger at oprette en ny generalisering fra klasse B til klasse A.

Et højreklik på en association viser en sammenhængsafhængig menu med handlinger som du kan anvende på den. Hvis du behøver at slette en association, vælges **Fjern** i den sammenhængsafhængige menu. Du kan også vælge **Egenskaber**, og afhængig af associationens type, redigere attributter såsom roller og mangfoldighed.

3.5.5.1 Ankerpunkter

Associationer tegnes, som standard, som en lige linje der forbinder de to objekter i diagrammet.

Du kan tilføje ankerpunkter til at bøje en association ved at dobbeltklikke et sted langs associationslinjen. Dette indsætter et ankerpunkt (som vises som et blåt punkt hvor associationslinjen er markeret), som du kan flytte omkring for at give associationen sin form.

Hvis du behøver fjerne en ankerpunkt, dobbeltklik på den igen for at fjerne den.

Bemærk at den eneste måde at redigere en associationsegenskab er via den sammenhængsafhængige menu. Hvis du forsøger at dobbeltklikke på den som med andre UML-objekter, indsættes blot et ankerpunkt.

3.5.6 Noter, tekst og felter

Noter, tekstlinjer og felter er elementer som kan findes i alle slags diagrammer, og har ingen virkelig semantisk værdi, men er meget hjælpsomme for at tilføje ekstra kommentarer eller forklaringer, som kan gøre diagrammet lettere at forstå.

For at tilføje en note eller tekstlinje, vælges tilsvarende værktøj i arbejdsværktøjslinjen, og så enkeltklikkes på diagrammet hvor du vil placere kommentaren. Du kan redigere teksten ved at åbne elementet via dets sammenhængsafhængige menu, eller for noter, også ved at dobbeltklikke på dem.

3.5.6.1 Ankre

Ankre bruges for til at linke noter og et andet UML-element sammen. Normalt bruger du for eksempel en note til at forklare eller give en kommentar om en klasse eller en vis association, og i så tilfælde kan du bruge ankeret til at gøre det klart at noten 'hører til' netop dette element.

Anvend ankerværktøjet i arbejdsværktøjslinjen, for at tilføje et anker mellem en note og et andet UML-element. Først skal du klikke på noten, og derefter klikke på UML-elementet som du vil at noten skal linkes til.

Kapitel 4

Kodeimport og kodegenerering

Umbrello UML Modeller er et UML-modelleringsværktøj, og som sådant er dets hovedformål at hjælpe dig med *analyse og konstruktion* af dine systemer. For at gøre overgangen fra konstruktion til *implementering* nemmere, tillader Umbrello UML Modeller dog at generere kildekode i forskellige programmeringssprog for at komme i gang. Hvis du desuden vil begynde på at bruge UML i et projekt som allerede er startet, kan Umbrello UML Modeller hjælpe dig med at oprette en model af systemet fra kildekoden ved at analysere den og importere klasserne som findes i den.

4.1 Kodegenerering

Umbrello UML Modeller kan generere kildekode for diverse programmeringssprog, baseret på din UML-model for at hjælpe dig med at komme i gang med implementeringen af projektet. Koden som laves består af klassedeklarationer, med metoder og attributter, så du kan 'udfylde tomrummet' ved at sørge for funktionerne i klassernes operationer.

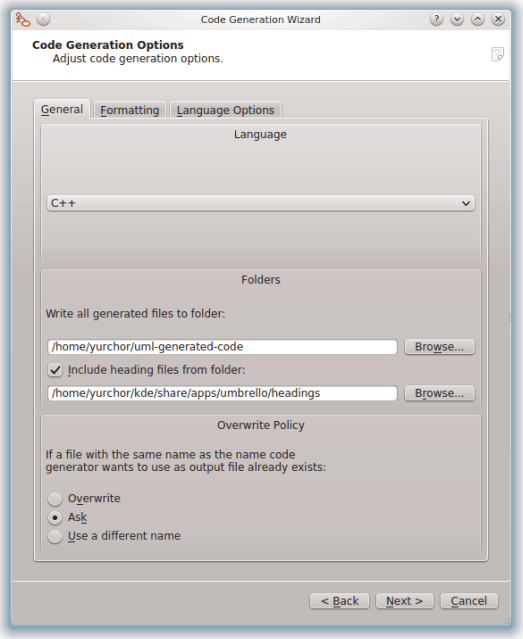
Umbrello UML Modeller 1.2 levereres med kodegenereringsunderstøttelse for ActionScript, Ada, C++, CORBA IDL, Java™, Javascript, PHP, Perl, Python, SQL and XML Schema.

4.1.1 Generér kode

For at generere kode med Umbrello UML Modeller, skal du først oprette eller indlæse en model som indeholder mindst en klasse. Når du er klar til at begynde at skrive lidt kode, vælges punktet **Kodegenereringsguide** i menuen **Kode**, for at starte guiden som leder dig gennem kodegenereringsprocessen.

Det første skridt er at vælge klasser, som du vil oprette kildekode for. Normalt vælges alle klasser i modellen, og du kan fjerne dem du ikke vil generere kode for, ved at flytte dem til listen på venstre side.

Næste skridt i guiden lader dig ændre parametre som kodegeneratoren bruger når den skriver koden. Følgende muligheder er til stede:



Valgmulighed for kodegenereringen i Umbrello UML Modeller

4.1.1.1 Kodegenereringsmulighed

4.1.1.1.1 Kodeinformationsniveau

Punktet **Skriv dokumenteringskommentarer selvom de er tomme** instruerer kodegeneratoren til at udskrive kommentarer i stilen /** blaha */, også selv om kommentarblokkene er tomme. Hvis du tilføjede dokumentation i klasser, metoder eller attributter til modellen, udskriver kodegeneratoren kommentarerne som Doxygen-dokumentation, uafhængig af hvad du angiver her, men hvis du vælger dette, udskriver Umbrello UML Modeller kommentarblokkene for alle klasser, metoder og attributter også selvom der ikke er nogen dokumentation i modellen, i hvilket tilfælde du bør dokumentere klasserne senere direkte i kildekoden.

Skriv kommentarer for afsnit selvom afsnittene er tomme: får Umbrello UML Modeller til at skrive kommentarer til kildekoden for at afgrænse de forskellige afsnit i en klasse. For eksempel 'Public methods' eller 'Attributes' før de tilsvarende afsnit. Hvis du vælger dette, så skriver Umbrello UML Modeller kommentarer for alle afsnit i klassen, også selvom afsnittet er tomt. Det ville for eksempel skrive en kommentar som lyder 'Protected methods', selvom der ikke er nogen sådanne i klassen.

4.1.1.1.2 Mapper

Skriv alle filer som laves til mappe: Her skal du vælge mappen hvor du ønsker at Umbrello UML Modeller skal lægge kildekoden som laves.

Punktet **Indsætte hovedfiler fra mappe**, lader dig indsætte et hovede i begyndelsen af hver fil som genereres. Hovedfiler kan indehold ophavsret- eller licensinformation, og kan indeholde variabler som evalueres når genereringen sker. Du kan tage et kig på skabeloner for hovedfiler som levereres med Umbrello UML Modeller, for at se hvordan man bruger variablerne til at erstatte dit navn eller dagens dato når genereringen sker.

4.1.1.1.3 Overskrivningspolitik

Dette fortæller Umbrello UML Modeller hvad der skal ske hvis filen som skal laves allerede findes i destinationsmappen. Umbrello UML Modeller 1.1 *kan ikke ændre eksisterende kildekodefiler*, så du skal vælge mellem at overskriver den eksisterende fil, springe over at oprette netop denne fil, eller lade Umbrello UML Modeller vælge et andet filnavn. Hvis du vælger at bruge et andet filnavn, tilføjer Umbrello UML Modeller en endelse til filnavnet.

4.1.1.1.4 Sprog

Umbrello UML Modeller genererer normalt kode for sproget som du har valgt som aktivt sprog, men du har mulighed for at ændre dette til et andet sprog med kodegenereringsguiden.

4.1.1.2 Generering med kodegenereringsguiden

Det tredje og sidste skridt i guiden viser status for kodegenereringsprocessen. Du behøver kun klikke på knappen Generér for at få klasserne udskrevet til dig.

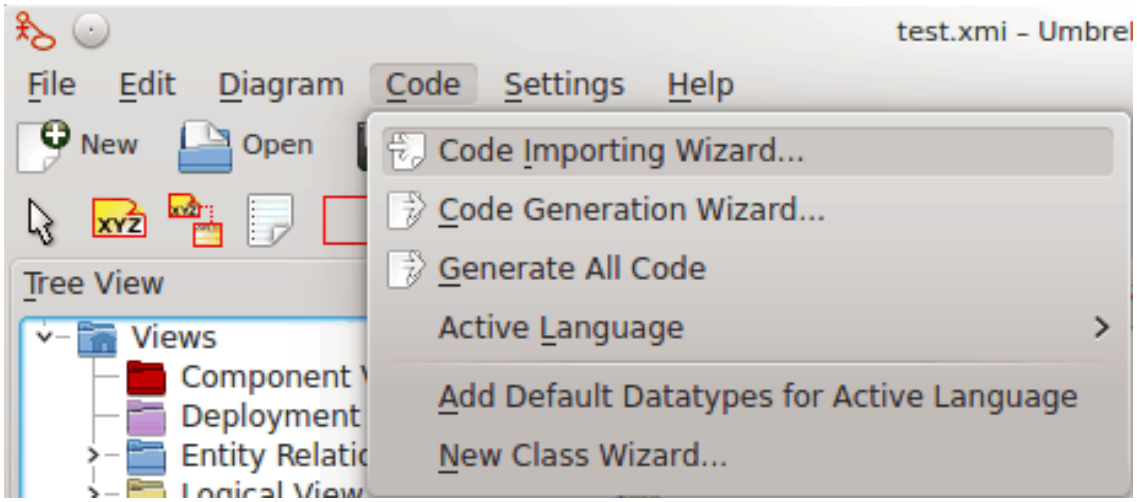
Bemærk at de tilvalg som du vælger med kodegenereringsguiden kun gælder for denne generering. Næste gang du kører guiden, skal du vælge alle tilvalg igen (din hovedmappe, overskrivningspolitik, og så videre). Du kan indstille standardværdier som bruges af Umbrello UML Modeller i afsnittet **Kodegenerering** i Umbrello UML Modellers indstillinger, tilgængelige via **Indstillinger** → **Indstil Umbrello UML Modeller...**.

Hvis du har indstillet kodegenereringstilvalg til de rigtige indstillinger, og vil lave lidt kode direkte uden at gå via guiden, kan du vælge **Generér al kode** i menuen Kode. Dette genererer kode for alle klasser i modellen med nuværende indstillinger (inklusive uddatamappe og overskrivningspolitik, så brug dette med forsigtighed).

4.2 Kodeimport

Umbrello UML Modeller kan importere kildekode fra eksisterende projekter for at hjælpe dig med at bygge modeller af systemer. Umbrello UML Modeller 1.2 understøtter kun C++ kildekode, men andre sprog vil være tilgængelige i fremtidige versioner.

For at importere klasser til modellen, vælges **Importér klasser...** i menuen **Kode**. Vælg filerne som indeholder C++ klassedeklarationer i fildialogen og tryk på O.k. Klasserne importeres, og du finder dem som en del af modellen i trævisningen. Bemærk at Umbrello UML Modeller ikke laver noget slags diagram for at vise klasserne, de importeres kun til modellen så du senere kan bruge dem i et valgfrit diagram.



Menu til at importere kildekode til Umbrello UML Modeller

Kapitel 5

Andre funktioner

5.1 Andre funktioner i Umbrello UML Modeller

Dette kapitel forklarer kortfattet nogle andre funktioner som Umbrello UML Modeller tilbyder.

5.1.1 Kopiér objekter som PNG-billeder

Foruden at tilbyde de normale funktioner for at kopiere, klippe og indsætte, som man kan forvente sig for at kopiere objekter mellem forskellige diagrammer, kan Umbrello UML Modeller kopiere objekter som PNG-billeder, så man kan indsætte dem i en hvilket som helst anden type dokument. Man behøver ikke gøre noget særligt for at bruge denne funktion, markér blot et objekt i et diagram (klasse, aktør, osv.) og kopiér det (**Ctrl-C**, eller brug menuen), åbn derefter et KWord-dokument (eller et andet program hvor billeder kan indsættes) og vælg **Indsæt**. Dette er en udmærket funktion for at eksportere dele af diagrammer som enkelte billeder.

5.1.2 Eksportere til et billede

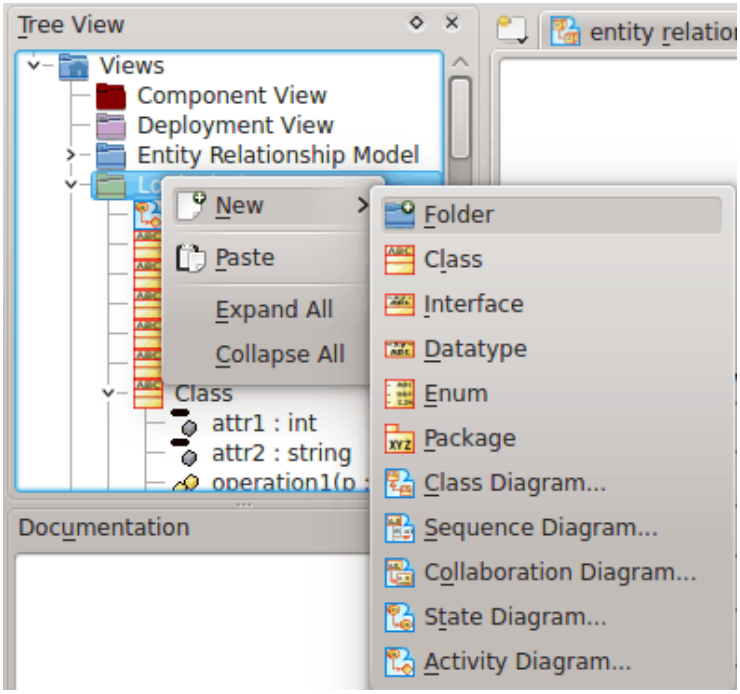
Man kan også eksportere et fuldstændigt diagram som et billede. Det eneste man skal gøre er at vælge diagrammet som skal eksporteres, og derefter punktet **Eksportér som billede...** i **Diagram**-menuen.

5.1.3 Udskrift

Umbrello UML Modeller tillader at enkelte diagrammer udskrives. Tryk på knappen **Udskriv** i programværktøjslinjen eller vælg punktet **Udskriv** i **Fil**-menuen, så vises KDE's standardudskriftsdialog hvor diagrammerne kan skrives ud.

5.1.4 Logiske mapper

For at organisere en model på en bedre måde, særligt for større projekter, kan man oprette logiske mapper i trævisningen. Vælg blot punktet **Ny** → **Mappe** i den sammenhængsafhængige menu i standardmappen under trævisningen, for at oprette dem. Mapper kan være indeni hinanden, og man kan flytte objekter rundt ved at trække dem fra en mappe og slippe dem i en anden.



Organisér en model med logiske mapper i Umbrello UML Modeller

Kapitel 6

Forfattere og historik

Dette projekt startedes af Paul Hensgen som et af hans universitetsprojekter. Det oprindelige navn på programmet var *UML Modeller*. Paul lavede al udvikling indtil slutningen af 2001, hvor programmet nåede version 1.0.

Version 1.0 tilbød allerede en hel del funktioner, men efter at projektet var blevet gennemset af Pauls universitet, kunne andre udviklere deltage, og de begyndte at give værdifulde bidrag til UML Modeller, såsom skift fra et binært filformat til en XML-fil, understøttelse af flere slags UML-diagrammer, kodegenerering og kodeimport, for kun at nævne nogle få.

Paul var tvunget til at trække sig tilbage fra udviklingsgruppen sommeren 2002, men som frit og åbent programmel, fortsætter programmet med at forbedres og udvikles, og vedligeholdes af en gruppe af udviklere fra forskellige dele af verden. Desuden ændrede projektet navn fra UML Modeller i september 2002. Der er flere grunde til navneændringen, den vigtigste at blot 'uml' som det var almindeligt kendt som, var et alt for generelt navn som forårsagede problemer med visse distributioner. En anden vigtig grund er at udviklerne synes at Umbrello er et meget smartere navn.

Udviklingen af Umbrello UML Modeller, samt diskussioner om i hvilken retning programmet skal udvikles i fremtidige versioner, er åben og finder sted via internettet. Hvis du skulle ville bidrage til projektet, så tøv ikke med at kontakte udviklerne. Der er mange måder du kan hjælpe Umbrello UML Modeller på:

- Rapportere fejl eller forbedringsforslag
- Rette fejl eller tilføje funktioner
- Skriv god dokumentation eller oversætte til andre sprog
- Og naturligvis... lav kode sammen med os!

Som du ser, er der mange måder du kan bidrage på. De er alle sammen meget vigtige og alle er velkomne til at deltage.

Umbrello UML Modeller-udviklerne kan nås på uml-devel@lists.sourceforge.net.

Kapitel 7

Ophavsret