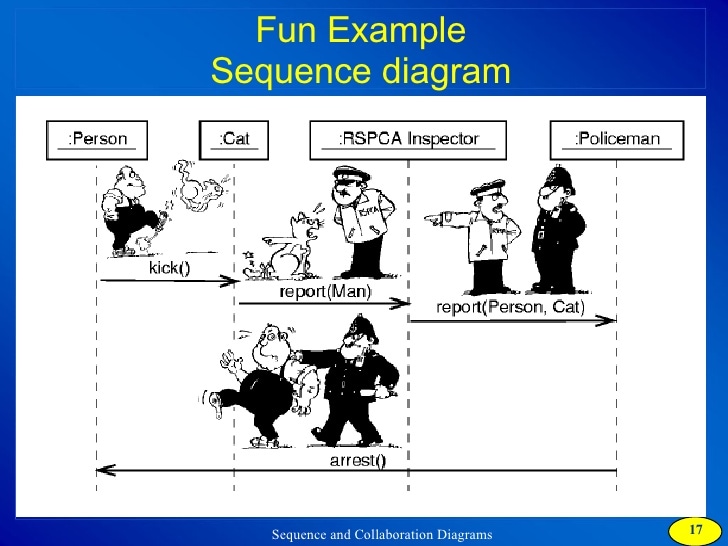
# Sequence Diagram And DCD

|  |  |
| --- | --- |
| **Læringsmål** | Du kan:   * **1Pf1**: anvende centrale metoder til at specificere og konstruere algoritmer [...] * **1Pf3**: anvende et i professionen udbredt, integreret udviklingsværktøj […] til at designe […] praksisnære applikationer baseret på en lagdelt arkitektur |
| **Forventet læringsudbytte [SOLO]** | Designsporet   * [Unistrukturel] Du kan genkende enkelte kendetegn ved **strukturmodellering (en: modeling structure)** – designklassediagram (DCD) * [Unistrukturel] Du kan genkende enkelte kendetegn ved **adfærdsmodellering (en: modeling behaviour)** – sekvensdiagram (SD) |
| **Din forberedelse** | Designsporet:   * [Larman] “Designing Objects: What are static and dynamic modeling?” (afsnit 14.4, side 216-217) * Genlæs/-se forberedelsen fra Ex08-SD-and-DCD:   + [Larman] ”UML Interaction Diagrams” (afsnit 15.1-15.4, side 221-238)   + [Larman] ”UML Class Diagrams” (afsnit 16.1-16.8, side 249-259)   + Se video om [Sequence and communication diagram](https://www.linkedin.com/learning/software-design-modeling-with-uml/sequence-and-communication-diagram) (til og med 7:08, se bort fra resten) * Se på Tusindfryd analyse-artefakter (OM, DM, UC, SSD, OC) **udarbejdet af underviserne** – De er i dokumentet ”*Tusindfryd artefakter samlet.docx*” på ItsLearning i Case-mappen for Tusindfryd-casen |

Du skal i denne opgave udarbejde et designklassediagram (DCD) fra en domænemodel (DM) samt udarbejde sekvensdiagrammer (SD) ud fra givne operationskontrakter (OC).

# Dagens ord:



# Øvelse 1: Terminologi

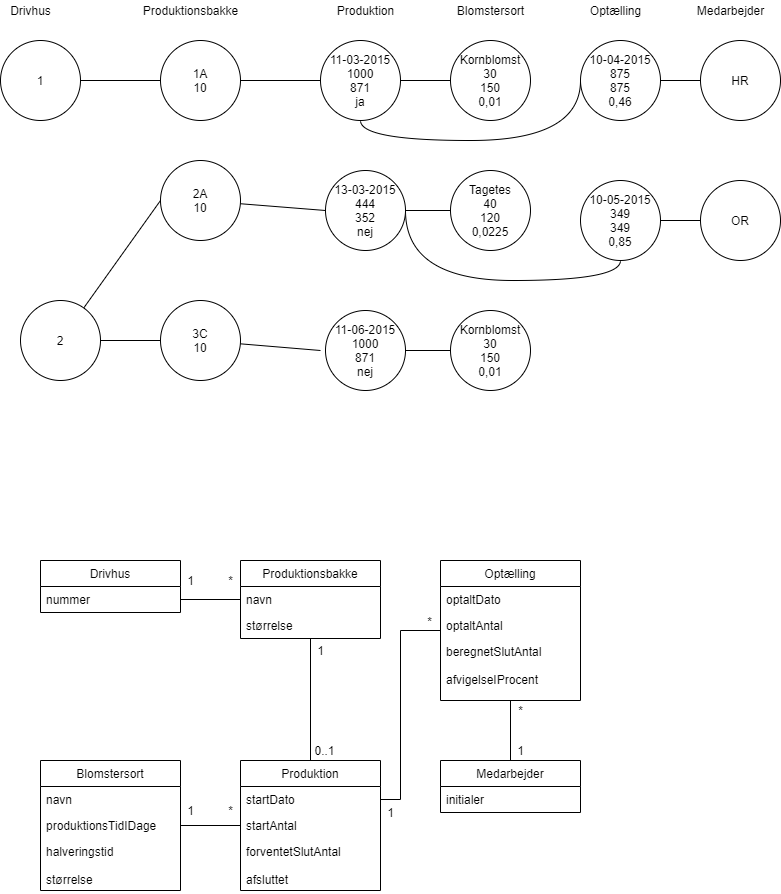
Del teamet op i to mindre grupper, og brug **Møde på midten** i hver gruppe til at diskutere begreberne: ”Static Modeling”, ”Dynamic Modeling”, ”designklassediagram” og ”sekvensdiagram”.   
Husk, at alle i gruppen skal have taletid.

# Øvelse 2: DCD for Tusindfryd

**Bemærk:** Du har i systemudvikling udarbejdet en række af artefakter for Tusindfryd, men for at sikre et fælles ståsted, så underviserne bedre kan facilitere dig samt for at sikre, at du kan sætte dig ind i artefakter udarbejdet af andre, skal du nu arbejde videre med undervisernes artefakter i dokumentet ”*Tusindfryd artefakter samlet.docx*” fra forberedelsen.

## Øvelse 2.1: Inspektion af Tusindfryds domænemodel

Inden du skal i gang med selve designklassediagrammet, skal du sikre dig, at du forstår undervisernes udgave af domænemodellen. Benyt derfor et par minutter på at inspicere og sammenligne den med domænemodel, du og dit team udarbejdede tidligere:



## Øvelse 2.2: Kvalitetskriterier til at tjekke DCD

Brug **møde på midten** i dit team til at lave en liste over kvalitetskriterier til at tjekke/kvalitetssikre designklassediagrammer og deres sporbarhed til andre relaterede artefakter. Jeres whiteboard skal fungere som midten (i CL-strukturen), så I laver en kolonne i den ene side af tavlen, hvor I øverst skriver alle jeres kvalitetskriterier op.

## Øvelse 2.3: Udarbejd et DCD

Med udgangspunkt i ovenstående domænemodel og de kvalitetskriterier, du og dit team har fundet, skal I nu sammen udarbejde et DCD på jeres whiteboard, hvor der dog også er et par andre krav, som I skal inddrage i udformningen:

* Du skal benytte dig af lagdeling, men tag kun højde for applikations- og domænelaget for nu (UI-laget kommer vi til i næste uge).
  + Se øvelse 3 i *Ex18-Disaheim4.docx* for inspiration, hvis du har behov for det.
* Der skal benyttes en Controller
* Der skal benyttes mindst et Repository

Overvej følgende undervejs:

* Hvorfor er det relevant med lagdeling?
* Hvilket problem løser controlleren?
* Hvorfor er det smart at benytte sig af et repository? Er det nok med et?
* Hvilke fagtermer er relateret til et DCD? (benyt dem gerne!)

# Øvelse 3: Sekvensdiagrammer for Tusindfryd

Et sekvensdiagram er det centrale diagram til at illustrere, hvilket ansvar du tildeler til de forskellige klasser (angivet i DCD’et) for afvikling af en bestemt operation i dit it-system. Hver klasses ansvar illustreres i form af de beskeder (en: message), som instanser af dine klasser udveksler mellem hinanden henover et tidsforløb. Selvom beskeder mellem objekter i sidste ende bliver implementeret som metode­kald i C#, så giver sekvensdiagrammer dig mulighed for at abstrahere helt fra et konkret program­merings­sprog og er dermed også et velegnet værktøj til at beskrive det algoritmiske design (trin, sekvenser, valg, gentagelser, betingelser) fra Computational Thinking.

## Øvelse 3.1: Kvalitetskriterier til at tjekke SD’er

Brug **møde på midten** til at lave en liste over kvalitetskriterier til at tjekke/kvalitetssikre sekvensdiagrammer (**bemærk:** ikke et SSD) og deres sporbarhed til andre relaterede artefakter. Jeres whiteboard skal igen fungere som midten (i CL-strukturen), så I skriver alle jeres kvalitetskriterier for SD’er op i modsatte side af jeres kriterier for DCD’er.

## Øvelse 3.2: SD ud fra OC

Udfør følgende:

* Du og dit team deler sig i to grupper.
* Fordel de forskellige operationskontrakter mellem de to grupper
* Med udgangspunkt i de valgte operationskontrakter og den fælles udarbejdede DCD (fra øvelse 2), da udarbejd i hver gruppe så mange SD’er, I kan inden for den givne tidsramme.
  + Det er ok, hvis I ikke når så mange, da I skal arbejde videre med Tusindfryd-casen i følgende opgaver

# Øvelse 4 (kl. 10.50): Indsaml viden om DCD og SD’er fra de andres whiteboards

Udfør følgende:

* Hver enkelt i teamet går ud til de andre teams og kigger på deres tavle (husk noget at skrive med og på). Se efter, om der er noget, du har overset, kvalitetskriterier, tegneteknikker, mm.
* Bring den viden, I har fået, med tilbage til jeres eget team.
* Opdatér jeres kvalitetskriterier, DCD og SD’er med den indsamlede viden

*Tidsramme: 20 minutter*

# Opsummering (kl. 11.10)