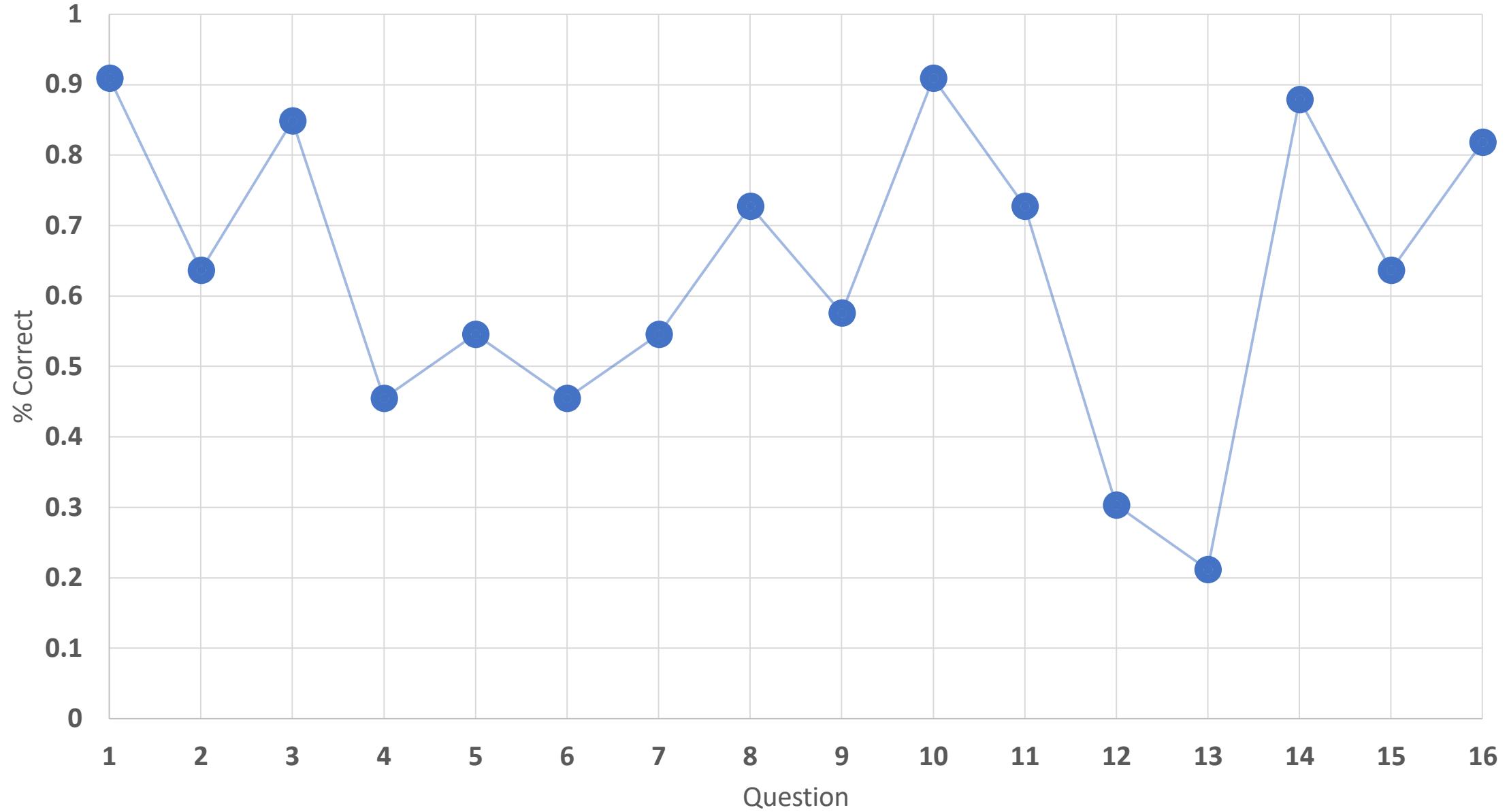


A man with dark hair and glasses, wearing a leather jacket over a white shirt, stands in a cluttered office. He is looking upwards and slightly to his right. The background is filled with stacks of papers, filing cabinets, and computer monitors. The title "HACKERMAN" is overlaid at the bottom in large, colorful, block letters.

HACKERMAN



1 - Quiz

UP Står for



30 sec



Unified process



Unified publication



Unified population



Unified procrastination



2 - Quiz

UML Står for



30 sec



Unification Modeling Language

✗



United Modeling Language

✗



Unified Modeling lingo

✗



Unified Modeling Language

✓

3 - Quiz

Hvad betyder risiko drevet?



30 sec



Lav risiko elementer udvikles først.



Elementer med høj risiko udvikles først.



Teamet laver så lidt som muligt for at holde spændingen oppe



Høj risiko elementer udvikles slet ikke.



4 - Quiz

Hvad betyder klient drevet?



30 sec

- Elementer med størst værdi for kunden udvikles først. ✗
- Systemet udvikles i samarbejde med kunden. ✗
- Alle før nævnte muligheder. ✓

5 - Quiz

Formålet med systemudvikling (analyse og design) er

at opnå et stabilt grundlæggende udviklingsprincippet, der er en del af den vigtige teknologi for udviklingen af systemer.

Inden:

30 sec



A



B



C



D



- A) At skabe overblik over krav til systemet og at etablere grundlag for systemets realisering.
- B) At konstruere alle artifacts der er en del af den valgte udviklingsprocess.
- C) At lave dokumentation for alle detaljer af systemet.
- D) At lave nogle modeller til kunden.

“**Analysis** emphasizes an *investigation* of the problem and requirements, rather than a solution.”

- Applying UML and Patterns (Kap. 1, s. 6), Craig Larman.

“During **object-oriented analysis**, there is an emphasis on finding and describing the objects—or concepts—in the problem domain.”

- Applying UML and Patterns (Kap. 1, s. 7), Craig Larman.

6 - Quiz

Formålet med inception fasen er



30 sec



At danne sig et fuldt overblik over projektet.



At undersøge alle projektets krav.



At udvikle alle analyse artefakter.



At se om projektet er gennemført.



“However, **the purpose of the inception step is not to define all the requirements**, or generate a believable estimate or project plan. At the risk of over-simplification, the idea is to do just enough investigation to form a rational, justifiable opinion of the overall purpose and feasibility of the potential new system, and decide if it is worthwhile to invest in deeper exploration (the purpose of the elaboration phase).”

- Applying UML and Patterns (Kap. 4, s. 48), Craig Larman.

7 - Quiz

Hvilke af disse er et ikke funktionelt krav



30 sec

- Systemet skal kunne håndtere æbler. ✗
- Systemet skal kunne håndtere maling. ✗
- Systemet skal have en gennemsnitlig svartid på 2 sekunder ✓
- Systemet skal kunne beregne rabat ✗

FURPS

- **Functional**
 - Features, capabilities, security.
- **Usability**
 - Human factors, help, documentation.
- **Reliability**
 - Frequency of failure, recoverability, predictability.
- **Performance**
 - Response times, throughput, accuracy, availability, resource usage.
- **Supportability**
 - Adaptability, maintainability, internationalization, configurability.

“In common usage, requirements are categorized as **functional** (behavioral) or **non-functional** (everything else);”
- Applying UML and Patterns (Kap. 5, s. 57), Craig Larman.

8 - Quiz

F i FURPS står for..



30 sec



Frequency

✗



Feasibility

✗



Functionality
(Functional)

✓



Faulty

✗

9 - Quiz

Formålet med et rigt billede er at vise dataflow



20 sec



Sandt



Falskt



"En metode til at opnå et domæne kendskab er tegne et rigt billede i samarbejde med en person med kendskab til domænet. Et rigt billede er en tegning uden formelle symboler eller egentlig syntaks og det bruges til at skabe et visuelt billede overblik over hvilke enheder, processer, problemer og strukturer der er i domænet."

- https://cloud.cct.au.dk/wiki/index.php?title=Rige_billeder

10 - Quiz

Hvilken er ikke en usecase type



30 sec



Concise



Brief



Casual



Fully dressed



- **brief**—terse one-paragraph summary, usually of the main success scenario.
The prior *Process Sale* example was brief.
 - **casual**—informal paragraph format. Multiple paragraphs that cover various scenarios. The prior *Handle Returns* example was casual.
 - **fully dressed**—the most elaborate. All steps and variations are written in detail, and there are supporting sections, such as preconditions and success guarantees.
- Applying UML and Patterns (Kap. 6, s. 66), Craig Larman.

11 - Quiz

En aktør kan befinde sig inde i ens system?



30 sec

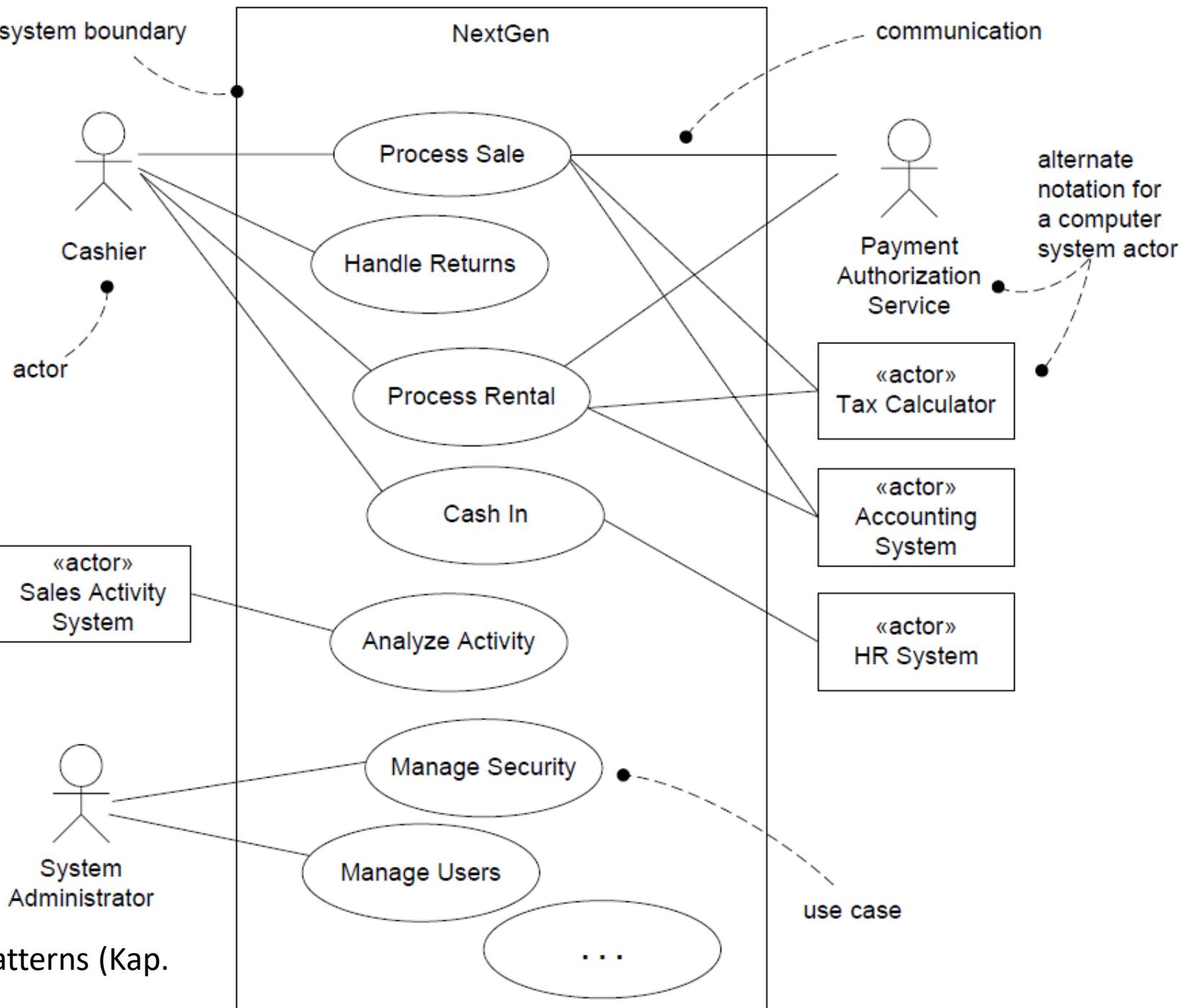


Sandt



Falsk





12 - Quiz

**Requirements traceability matrix viser sammenhængen
mellem..**



30 sec



Krav og aktør

✗



Krav og interresenter

✗



Krav og use cases

✓



Aktør og use cases

✗

Sammenhæng mellem krav og use cases

Der er mange-til-mange relationer mellem krav og use cases

- En use case kan dække over flere funktionelle krav
- Et funktionelt krav kan realiseres af flere use cases

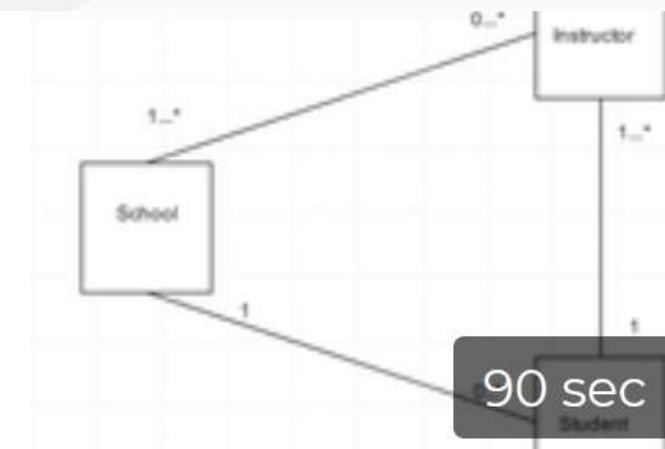
Arlow & Neustadt:

Requirements	Use cases				
		U1	U2	U3	U4
R1					
R2					
R3					
R4					
R5					

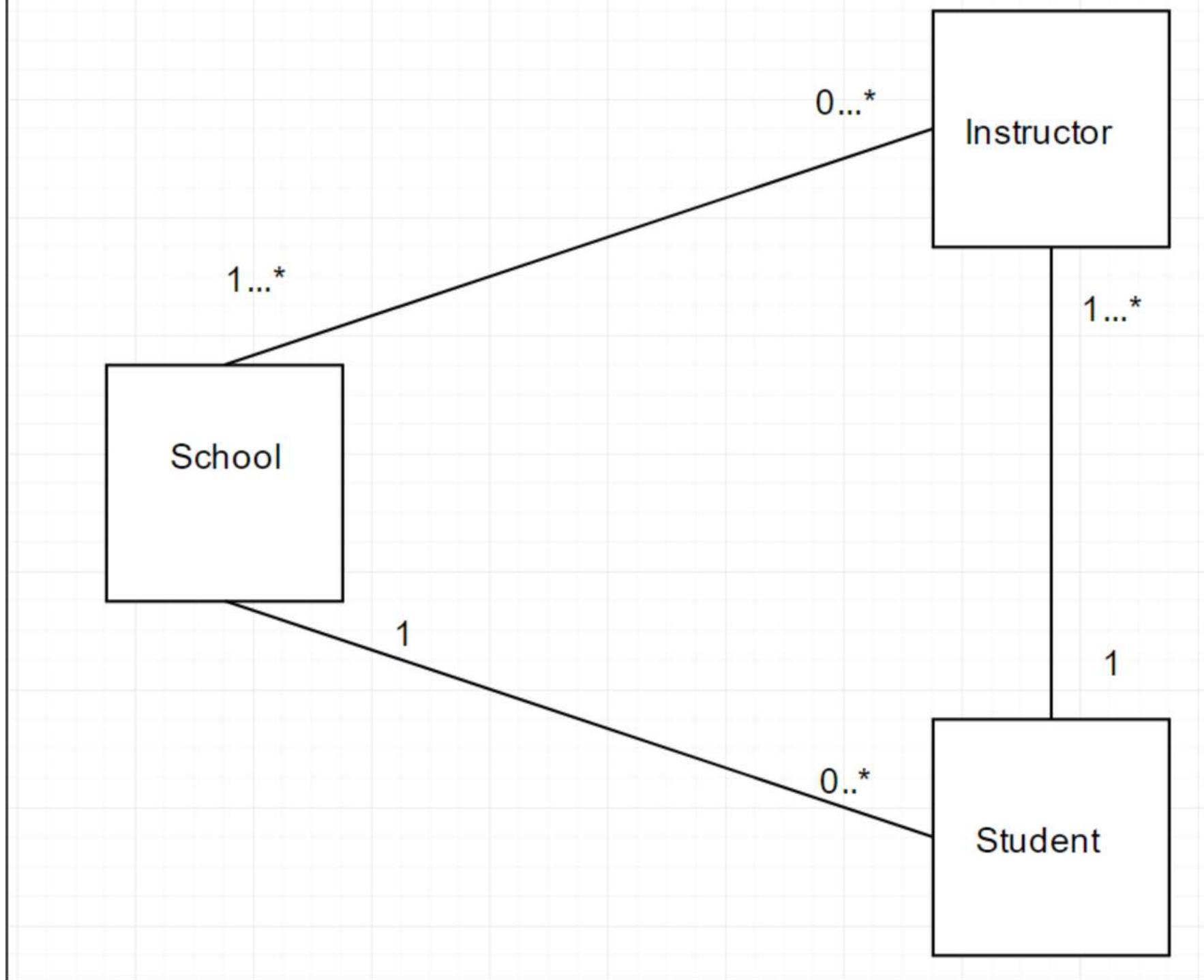
Requirements
Traceability
Matrix

14 - Quiz

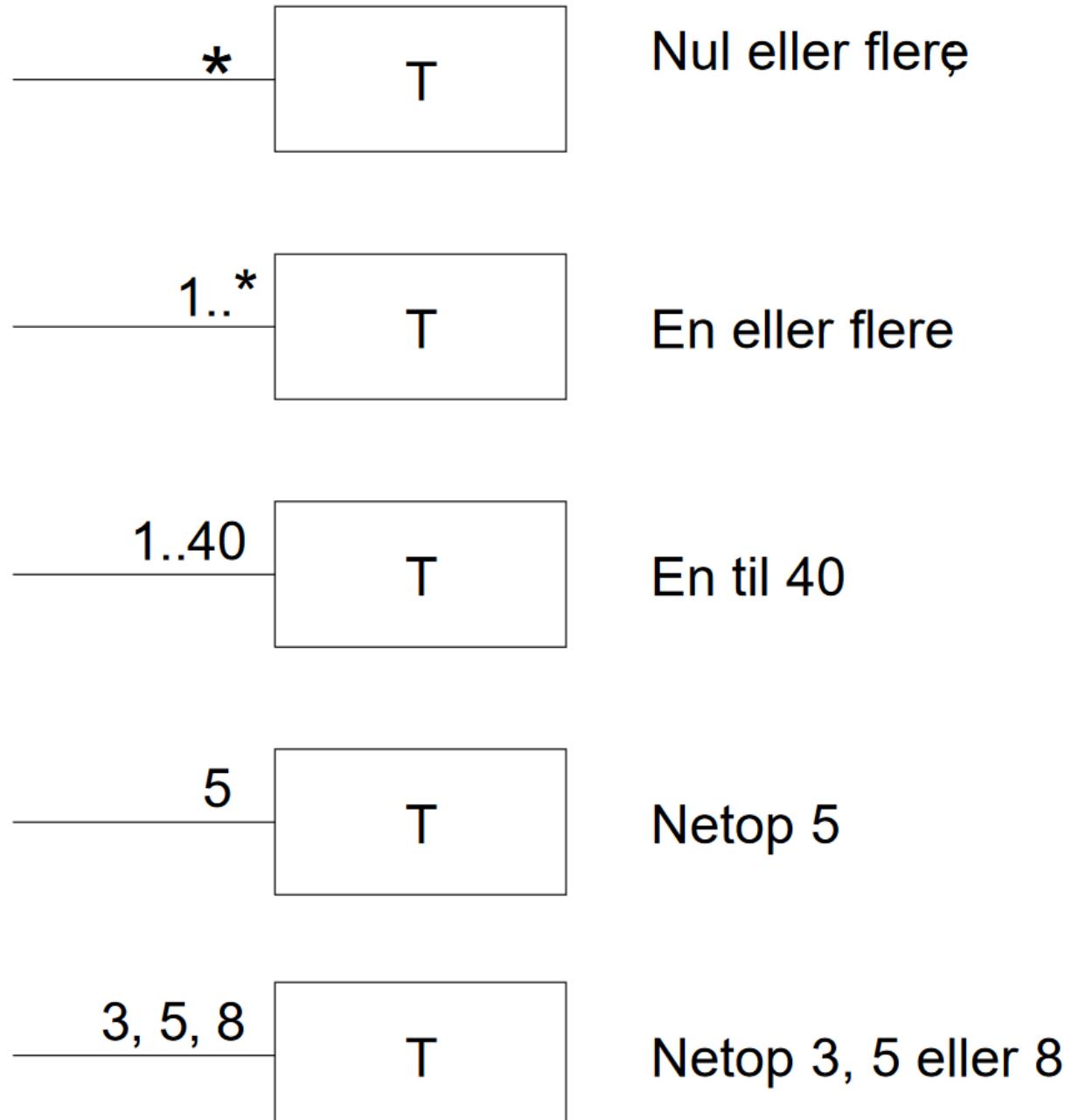
En studerende er tilknyttet

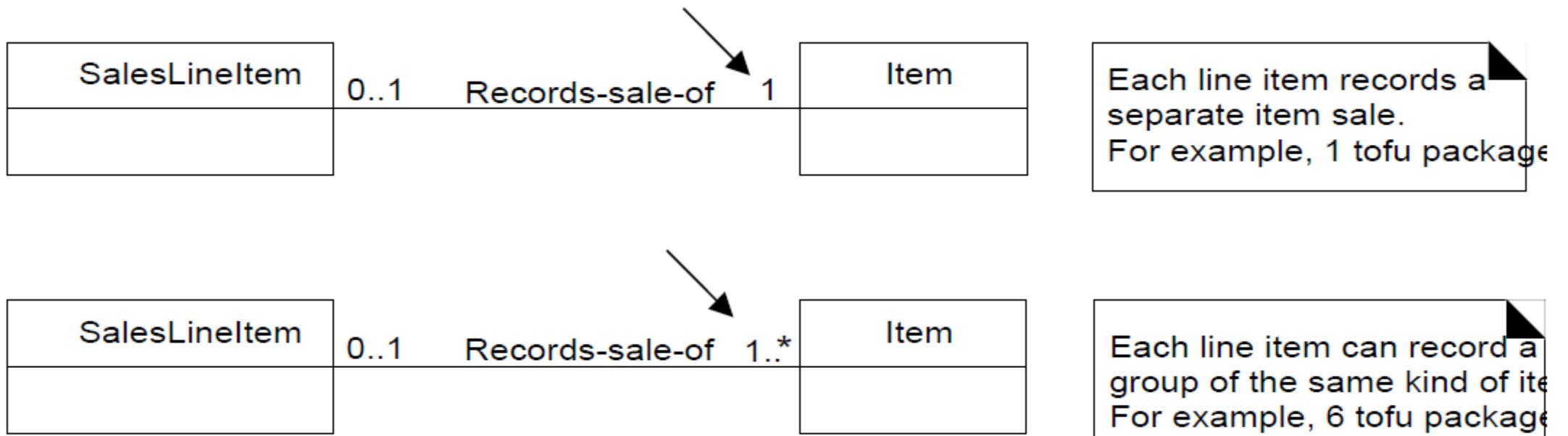


- 1 til mange instructors ✓
- 1 instructor ✗
- 0 til mange skoler ✗

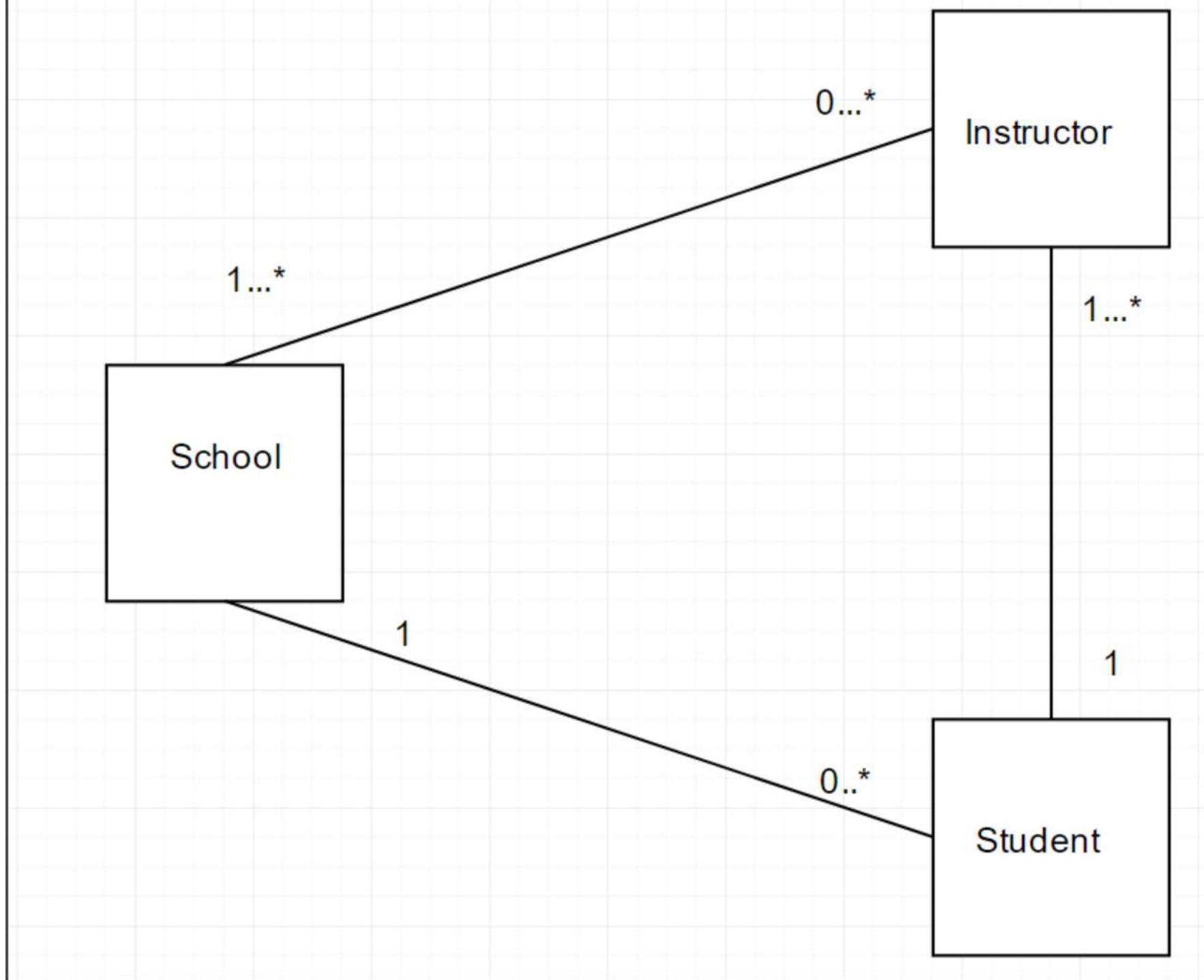


Larman fig. 9.14:





- Applying UML and Patterns (Kap. 9, s. 154), Craig Larman.



15 - Quiz

En skole kan have "X" TA's



90 sec



En

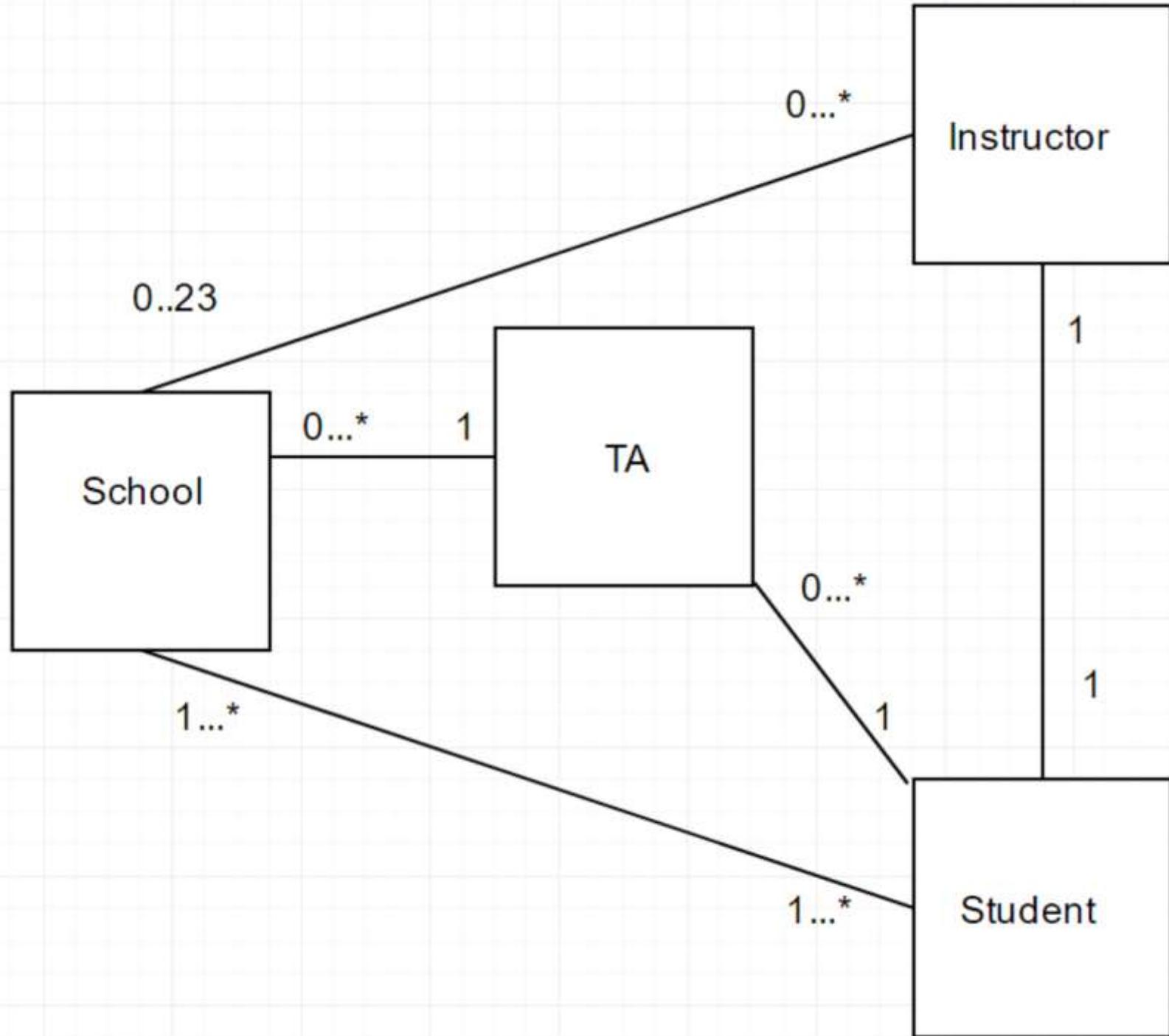


Nul til mange



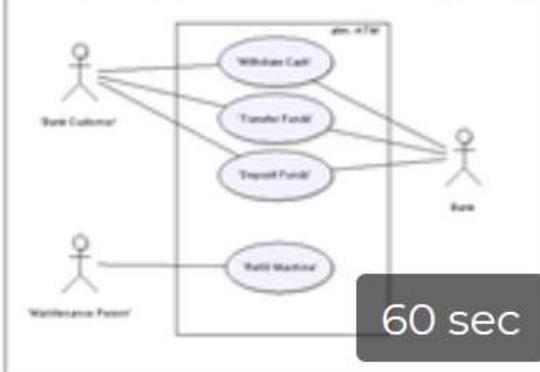
En til mange





16 - Quiz

Bank er en primær aktør



60 sec



Falsk

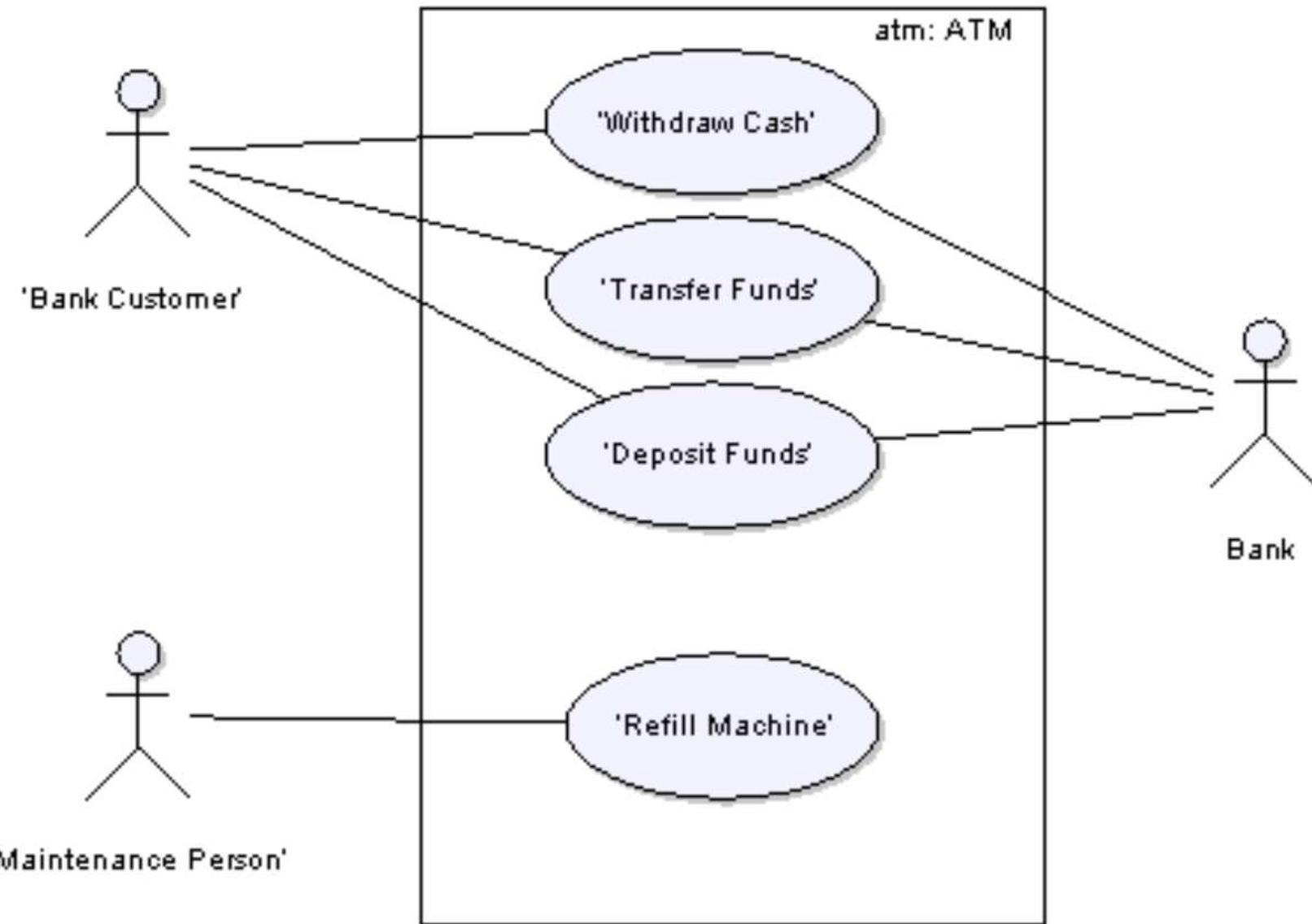


Sandt



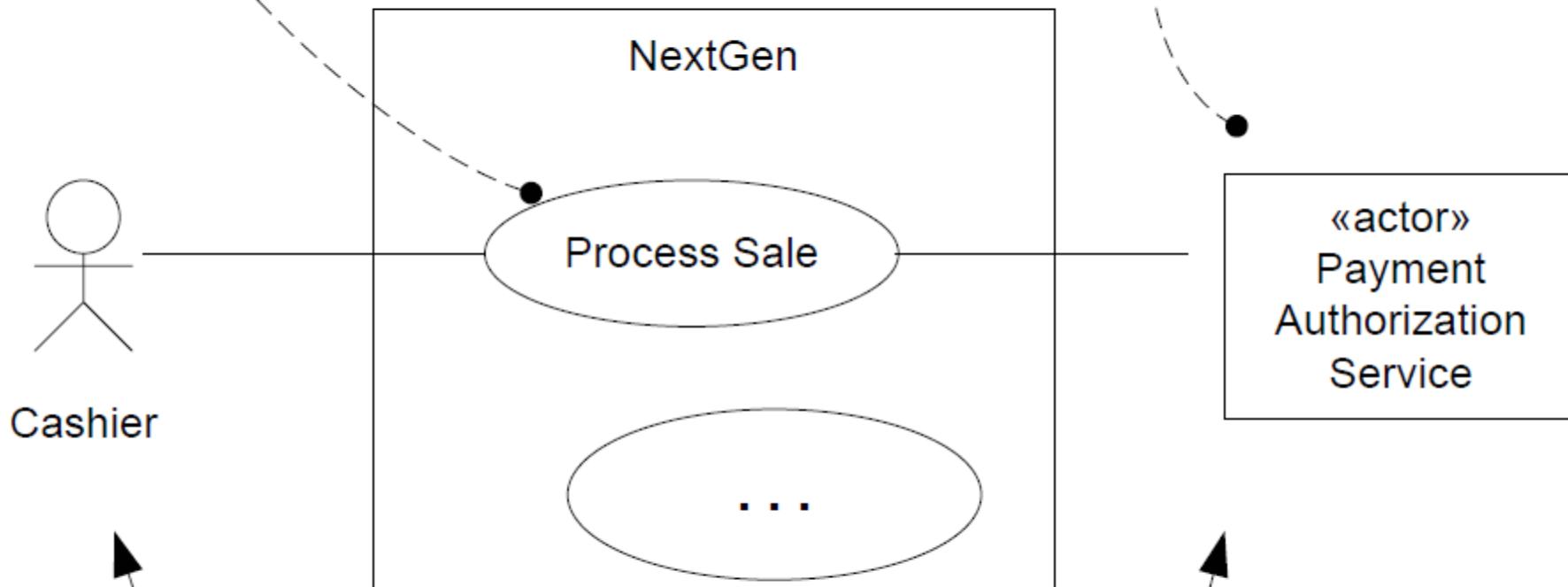
ATM Use Cases Inception

package OpenUP {1/2}



For a use case context diagram, limit the use cases to user-goal level use cases.

Show computer system actors with an alternate notation to human actors.

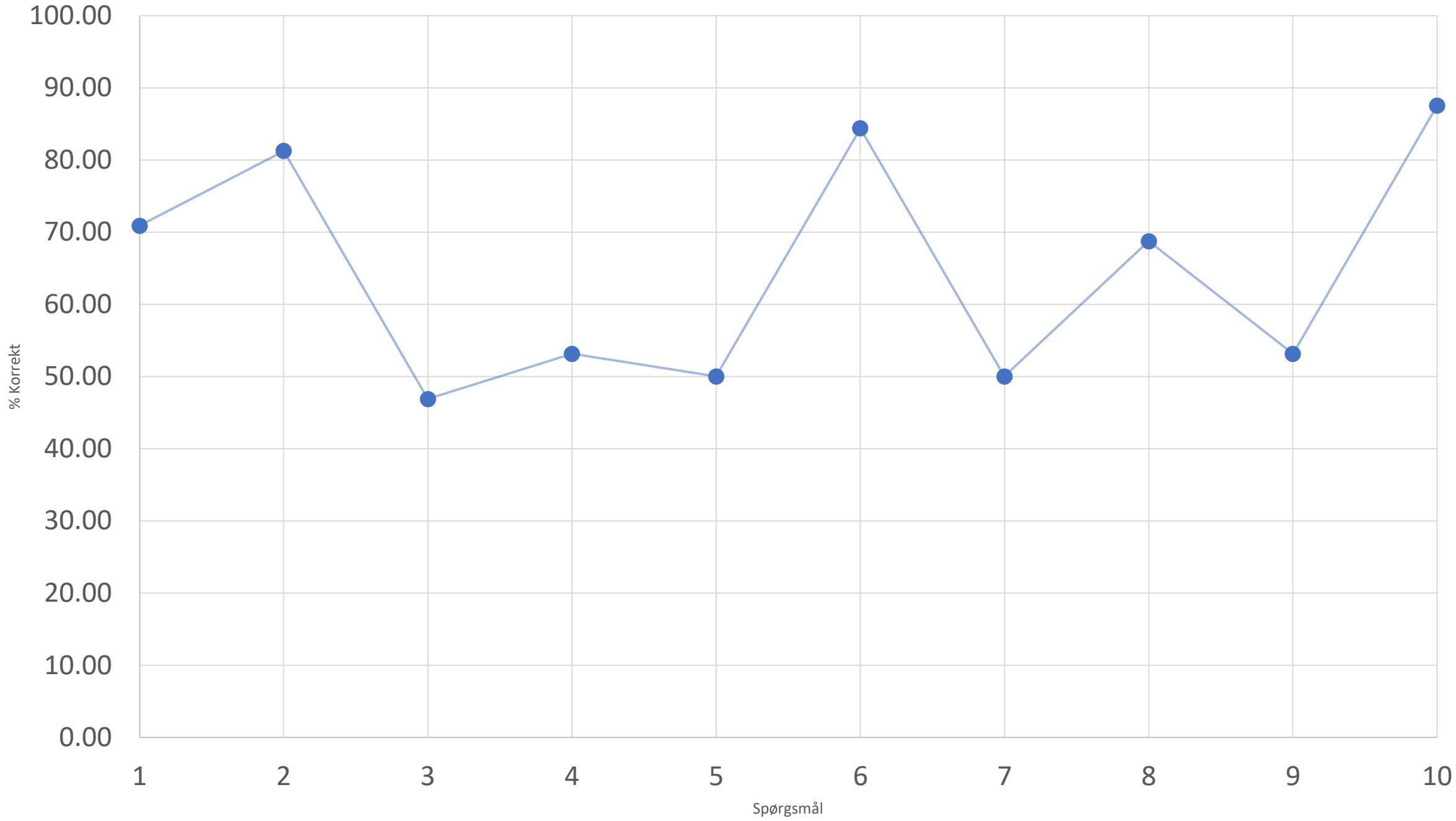


primary actors on
the left

supporting actors
on the right

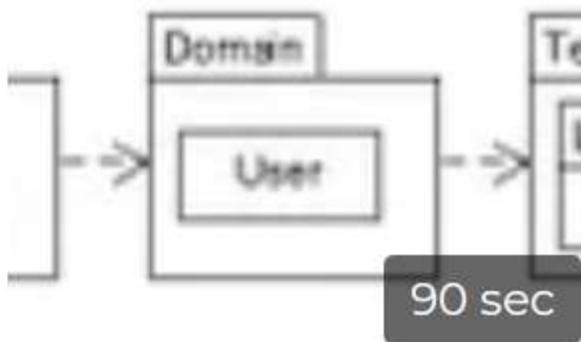


Kahoot

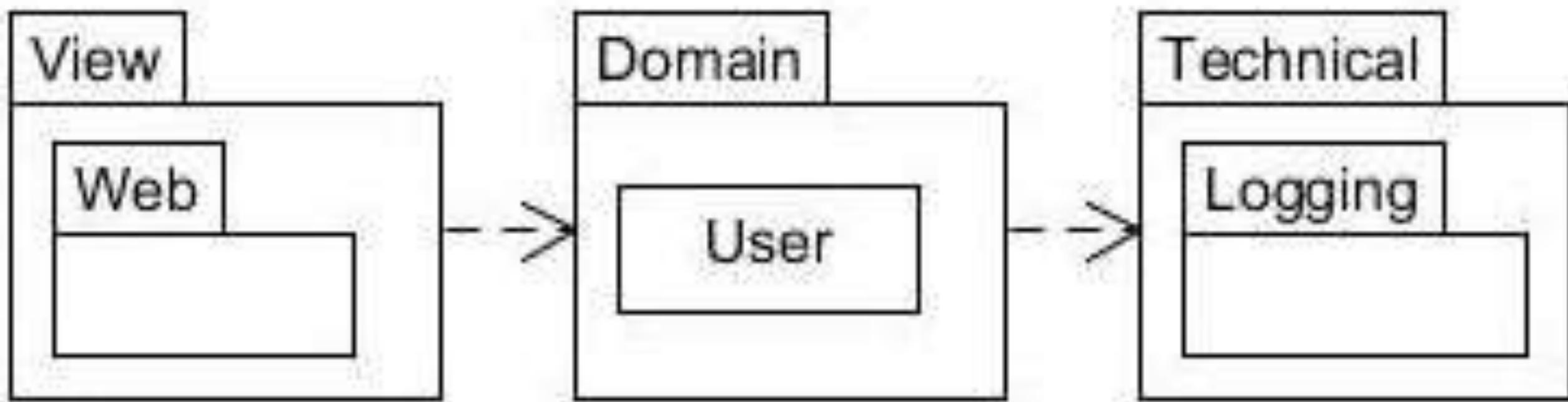


1 - Quiz

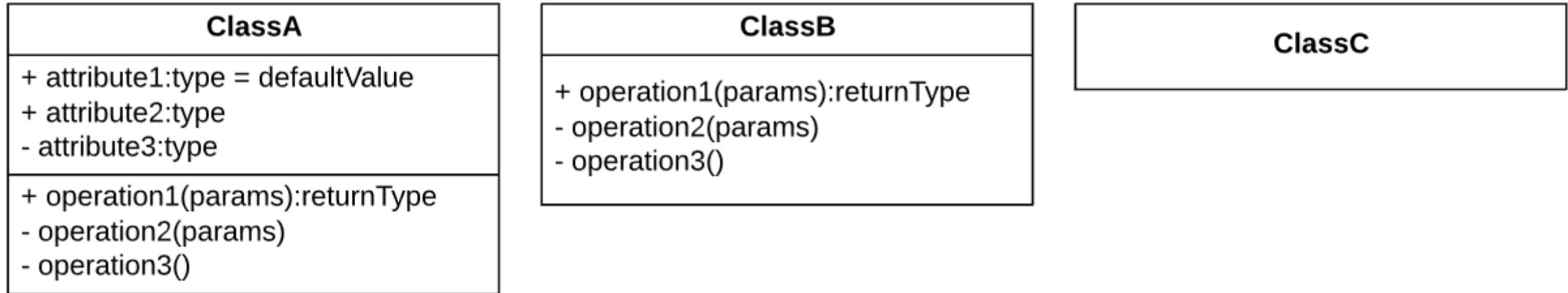
Hvad er User og Web i dette pakkediagram?



- Web er en klasse, User er en pakke X
- Web er en pakke, User er en pakke X
- Web er en pakke, User er en klasse ✓
- Web er en klasse, User er en klasse X

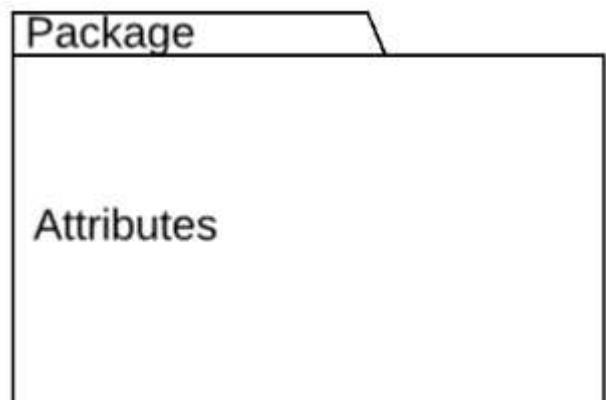


Klasser

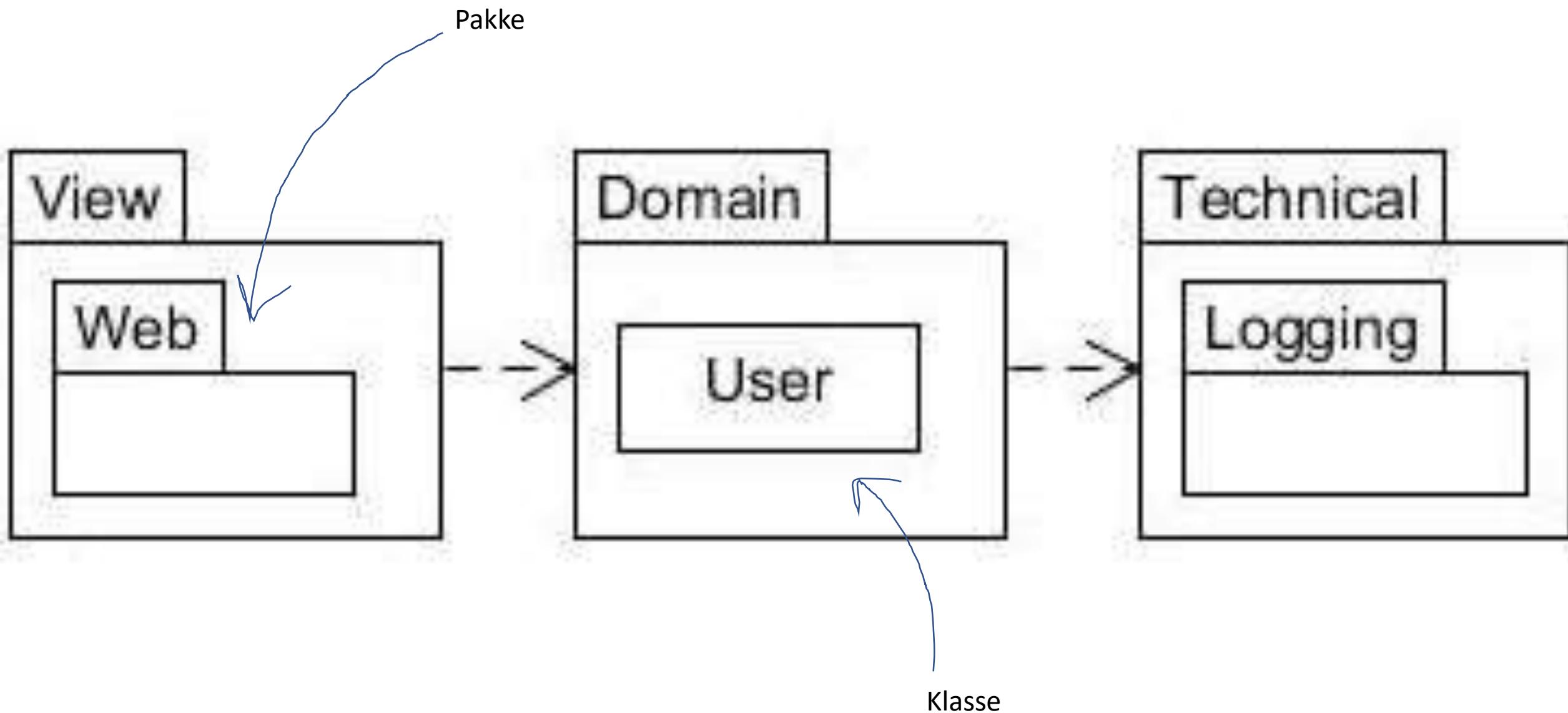


- Applying UML and Patterns, Craig Larman, Ch. 9

Pakke

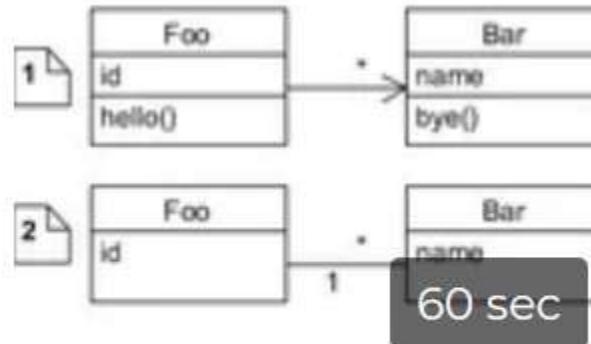


- Applying UML and Patterns, Craig Larman, Ch. 13

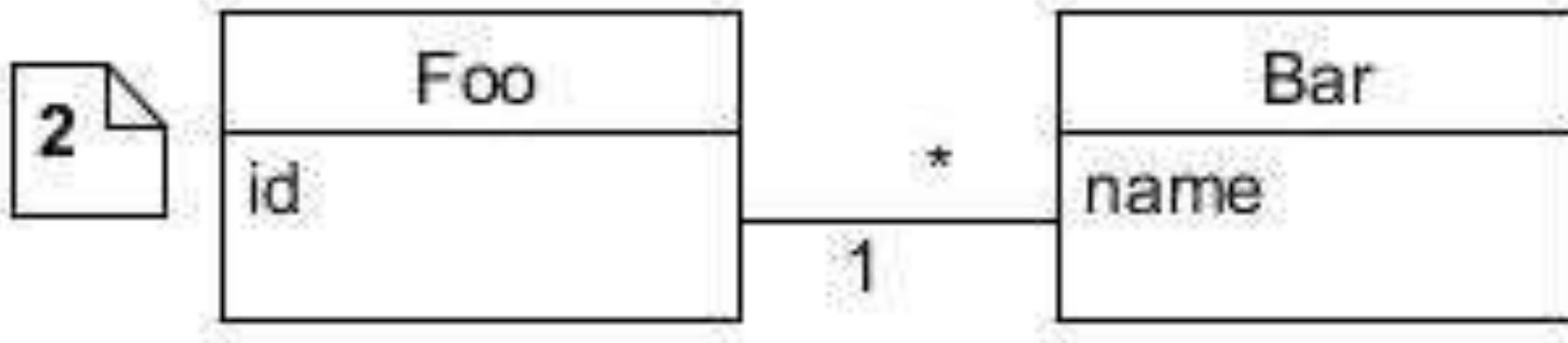
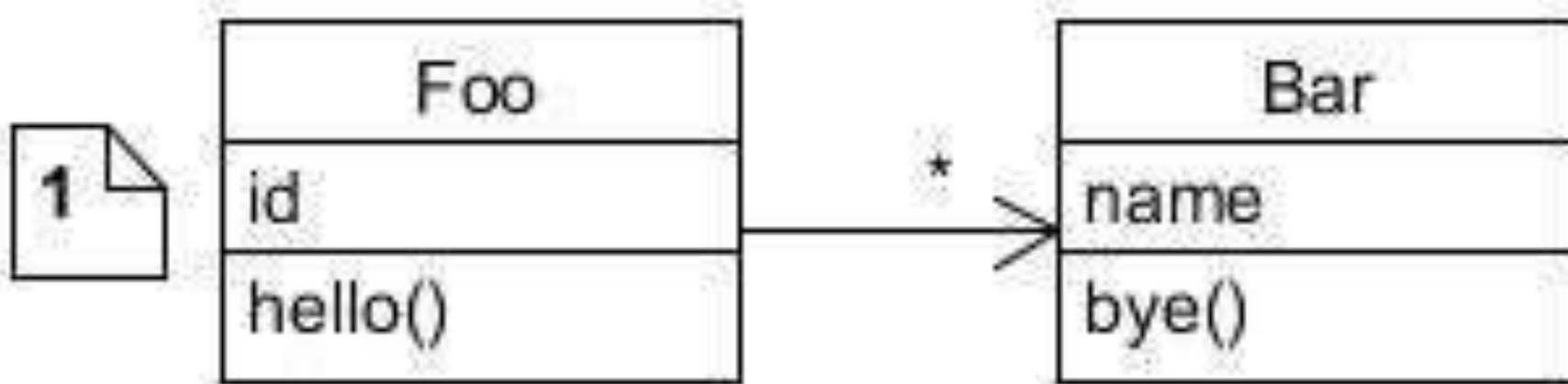


2 - Quiz

Hvad hedder disse UML diagrammer?



- 1 - Domænemodel, 2 - Design-klassediagram X
- 1 - Design-klassediagram, 2 - Domænemodel ✓
- 1 - Design-klassediagram, 2 - Design-klassediagram X
- 1 - Domænemodel, 2 - Domænemodel X



3 - True or False

Man skal altid have en aktør i et sekvensdiagram



60 sec



False

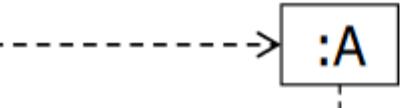


True



“The UML uncludes **interaction diagrams** to illustrate how objects interact via messages.”

- Applying UML and Patterns, Craig Larman, Ch. 15 p. 221

	Type meddelelse	Beskrivelse
	Synkron meddelelse	En operation kaldes synkront Afsender venter til modtageren har færdigbehandlet operationen
	Asynkron meddelelse	En operation kaldes asynkront Afsender venter IKKE til modtageren har færdigbehandlet operationen
	Meddelelse retur	Returnering fra et synkront operationskald Modtageren returnerer kontrollen til afsenderen
 	Opret	Afsender opretter et objekt
	Nedlæg	Afsender nedlægger et objekt
	Meddelelse "Udefra"	Meddelelsen er sendt ude fra til interaktionsdiagrammet “A found message whose sender will not be specified.”
	Tabt meddelelse	Meddelelsen fejler og når ikke sit mål

4 - Quiz

Hvad er krav for polymorfi?



20 sec

- ▲ Superklassens attributter skal være public. ✗
- ◆ Nedarvede metoder skal have samme signatur som i superklassen. ✓
- Subklassen skal have samme navn som superklassen. ✗
- Subklassen må ikke definere nye metoder. ✗

Polymorfi

Polymorfi (mange former) kan kun anvendes i forbindelse med nedarvede klasser

- Der kan – på en ensartet måde - refereres til objekter af forskellige subklasser, der alle arver fra samme superklasse
- Nedarvede metoder skal have samme navne som i superklassen
- Metodekald til en superklasses metode vil på udførelsestidspunktet kalde den relevante metode fra en af subklasserne (sen binding)

5 - Quiz

Hvilken synlighed har NamedInput til Environment?

```
ass NamedInput extends Circuit{
    private String s;
    public String compile(Environment env){
        if(env.isDefined(s)){
            return s;
        } else {
            return null;
        }
    }
}
```

60 sec



Attribut



Parameter



Lokalt



```
]class NamedInput extends Circuit{  
    private String s;  
    public String compile(Environment env){  
        if(env.isDefined(s)){  
            return s;  
        } else {  
            return null;  
        }  
    }  
}
```

B er en attribut i A – dvs. A har adgang til B, når A holder en reference til B som attribut

Eksempel:

```
class A
{
    B objekt;
    ...
}
```

B er en parameter til en metode i A – dvs. A har adgang til B, når A får overført en reference til B

- Midlertidig synlighed
- Kan bliver permanent, hvis referencen gemmes

Eksempel:

```
class A
{
    ...
    public void MetodeA (B bobjekt)
    {
        bobjekt.MetodeB( );
        ...
    }
    ...
}
```

B er et lokalt objekt (en lokal variabel) i en metode i A

Kendskabet eksisterer kun inden for den givne metode

Eksempel:

```
class A
{
    ...
    public void MetodeA()
    {
        B objekt = new B();
        objekt.MetodeB();
        ...
    }
}
```

Hvilken synlighed har NamedInput til Environment?

```
]class NamedInput extends Circuit{  
    private String s;  
    public String compile(Environment env){  
        if(env.isDefined(s)){  
            return s;  
        } else {  
            return null;  
        }  
    }  
}
```

6 - Quiz

Hvilken af følgende muligheder er IKKE en del af GRASP?



30 sec

- Creator ✗
- Destroyer ✓
- Information ~~Export~~ Expert ✗
- Polymorphism ✗

General Responsibility Assignment Software Patterns

Contents [hide]

1 Patterns

- 1.1 Controller
- 1.2 Creator
- 1.3 Indirection
- 1.4 Information expert
- 1.5 High cohesion
- 1.6 Low coupling
- 1.7 Polymorphism
- 1.8 Protected variations
- 1.9 Pure fabrication

2 See also

3 Notes

4 References

[https://en.wikipedia.org/wiki/GRASP_\(object-oriented_design\)](https://en.wikipedia.org/wiki/GRASP_(object-oriented_design))

7 - True or False

En abstrakt klasse kan instantieres.



20 sec



False



True

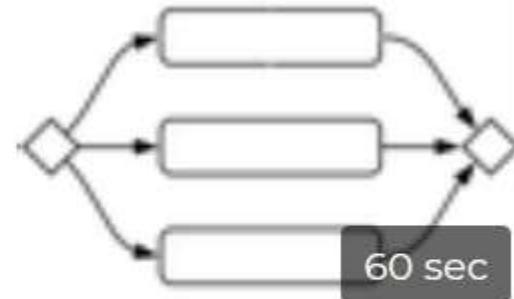


“A class which is declared as abstract is known as an **abstract class**. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.”

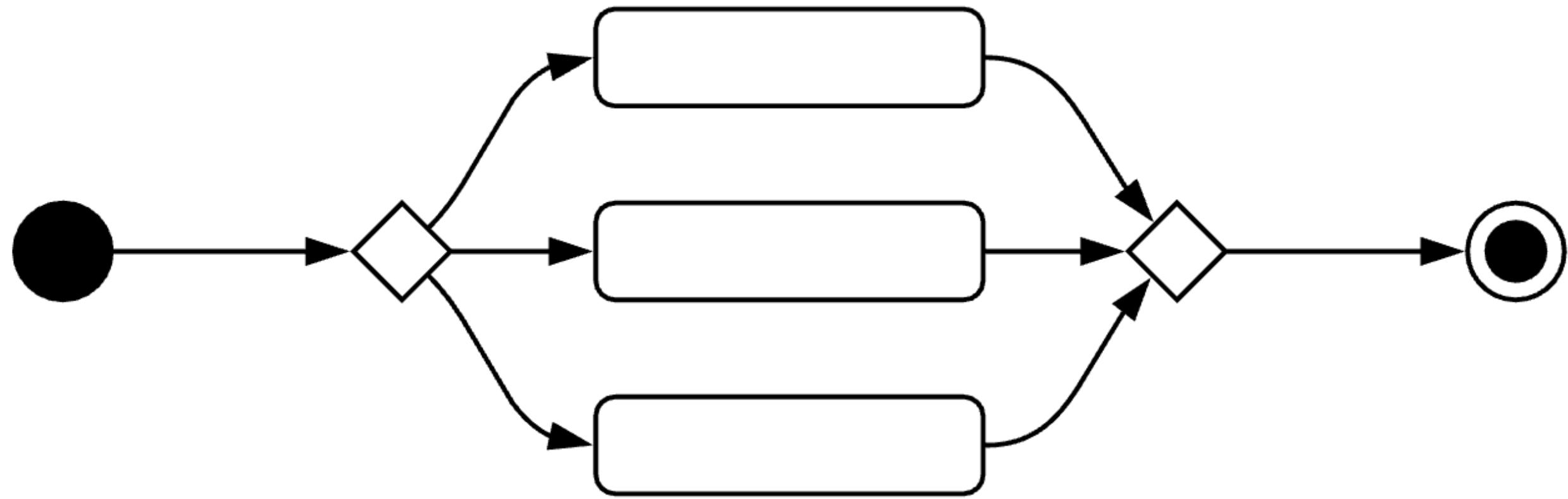
<https://www.javatpoint.com/abstract-class-in-java>

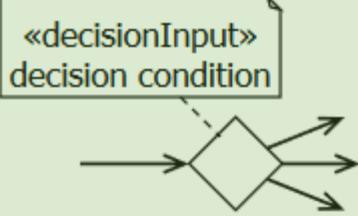
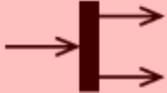
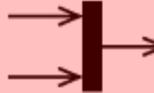
8 - Quiz

Hvad viser diagrammet?



- Et sekvensdiagram. ✗
- Aktivitetsdiagram med 3 aktiviteter der sker parallelt. ✗
- Klassediagram ✗
- Aktivitetsdiagram med en decision node der har 3 mulige udfald. ✓



Control node	Beskrivelse	
	Initial node – diagrammet starter her. Hvis der er flere, startes alle	Final nodes
	<u>Activity final</u> node – diagrammet slutter her (stopper alle flows)	
	<u>Flow final</u> node – stopper et specifikt flow – andre flows fortsætter	
	Decision node – der angives en betingelse på hver ud-pil	
	Merge node	
	Fork node – opdeler flow'et i flere parallelle flows	
	Join node – samler flere parallelle flows (logisk AND)	

9 - Quiz

Når vi designer klasser hvad ønsker vi så at opnå?



30 sec



Høj binding og høj kobling.



Lav binding og høj kobling.



Lav binding og lav kobling.



Høj binding og lav kobling.



Contents [hide]

1 Patterns

[1.1 Controller](#)

[1.2 Creator](#)

[1.3 Indirection](#)

[1.4 Information expert](#)

[1.5 High cohesion](#)

[1.6 Low coupling](#)

[1.7 Polymorphism](#)

[1.8 Protected variations](#)

[1.9 Pure fabrication](#)

2 See also

3 Notes

4 References

Et element har lav kobling, når det ikke er afhængig af, har kendskab til eller er forbundet med alt for mange andre elementer

Tildel ansvar, så kobling forbliver lav

Binding udtrykker den grad et objekts ansvar er homogent og sammenhængende

Tildel ansvar, så binding forbliver høj

Binding kan øges ved:

- Uddelegering af opgaver
- Opdeling af klasser

10 - Quiz

Hvad betyder risiko drevet?



30 sec

- Lav risiko elemnter udvikles først. X
- Elementer med høj risiko udvikles først. ✓
- Teamet laver så lidt som muligt for at holde spændingen oppe. X
- Høj risiko elementer udvikles slet ikke. X

Introduktion til systemudvikling

**Objektorienteret Analyse & Design
Unified Process
Kp. 1-3**

Opgave

Lektion 1.1

Følgende to principper skal uddybes:

- Risiko-drevet
 - Hvad betyder det?
 - Hvilke risici kan opstå? Nævn eksempler.
 - Kan risici forebygges? Evt. hvordan?
- Klient-drevet
 - Hvad betyder det?

Opgaver

Lektion 1.2

Hvilke procesorienterede aktiviteter skal foregå sideløbende med selve software-udviklingen?

Lektion 1.3

- Udarbejd en liste over de færdigheder og egenskaber en systemudvikler bør have. Alle punkter skal begrundes.
- Vurdér herefter, hvilke af disse færdigheder og egenskaber, du selv opfylder.
- Hvordan kan du opnå færdigheder i de punkter, du er "knap så god" til?

Opgave

Lektion 1.4

- Undersøg hvilke freeware-værktøjer der findes til udarbejdelse af UML-diagrammer.
- Vælg et af dem og udfør eksperimenter.

Lektion 1.5

I bogen nævnes vandfaldsmodellen. Udarbejd en liste over fordele og ulemper for vandfaldsmodellen og Unified Process.

Lektion 1.6

Læs oplægget til ugeopgaveprojektet omhyggeligt igennem. Beslut – sammen med din projektgruppe – hvilket delsystem, I vil arbejde med i efterårssemestret. Udarbejd en kort beskrivelse (maks en halv side).

Vigtigt vedrørende ugeopgaverne

- Hver projektgruppe skal aflevere én pdf-fil med løsninger til ugeopgaverne **senest den efterfølgende onsdag kl. 20.00**
- TA'erne giver skriftlig og mundtlig feedback den efterfølgende mandag i øvelsestimerne

Husk:

- Én studerende afleverer opgaven og inviterer de øvrige gruppemedlemmer (DTU Inside)
- Gruppens øvrige medlemmer skal herefter acceptere at være med i gruppeafleveringen (DTU Studenter-mail)

Kravanalyse

**Introduktion til Inception
Kravindsamlingsteknikker
Kp. 4-5**

Opgaver

Lektion 2.1

- Undersøg og giv en beskrivelse af, hvordan en interessentanalyse kan udføres. Brug f.eks. de foreslæde links under "Supplerende noter" i semesterplanen.
- Giv herefter en vurdering af, i hvor høj grad dette kan benyttes i jeres ugeopgaveprojekt. Hvilke elementer fra analysen vil I anvende?

Lektion 2.2

Giv en kort beskrivelse af hvilke "faldgruber" man specielt skal være opmærksom på i forbindelse med brug af:

- Spørgeskemaundersøgelser
- Interviews

Opgave

Lektion 2.3

Tag udgangspunkt i jeres ugeopgaveprojekt.

- Bliv enige om formålet med jeres kommende IT-system.
- Udarbejd rige billeder.
- Udarbejd en oversigt over det kommende systems interesserter.
- Udarbejd en liste med krav til systemet.
- Udarbejd en krav – interessennt-tabel.

Kravspecifikation

Use Cases
Kp. 6

Opgave

Lektion 3.1

Arbejd videre med ugeopgaveprojektet.

- Udarbejd en liste over det kommende systems aktører.
- Beskriv kort de enkelte aktørers formål.
- Angiv om de er primære eller supporterende.

Opgave

Lektion 3.2

Arbejd videre med ugeopgaveprojektet.

- Identificér use cases og udarbejd et use case diagram.
- Udarbejd derefter en aktørtabel.

Opgave

Lektion 3.3

Arbejd videre med ugeopgaveprojektet.

- Udarbejd en liste over use cases samt deres beskrivelser i en brief udgave. Giv hver use case et kort navn.
- Vælg dernæst en central use case og beskriv den casual.
- Arbejd videre med den samme use case og gør beskrivelsen fully dressed.

Opgave

Lektion 3.4

Arbejd videre med ugeopgaveprojektet.

- Kravene fra opgave 2.1 og de korte use case navne fra opgave 3.3 skal benyttes til at udarbejd en oversigt over sammenhængen mellem use cases og krav.

Artifacts til kravbeskrivelser

Flere kravbeskrivelser

Opsummering: Inception & elaboration

Kp. 7-8

Opgave

Lektion 4.1

Arbejd videre med ugeopgaveprojektet. Der skal påbegyndes udarbejdelse af følgende:

- Vision
- Supplerende specifikationer
- Forretningsregler/domæneregler
- Ordbog

Opgave

Lektion 4.2

- Undersøg og giv en beskrivelse af, hvordan en risikoanalyse kan udføres. Brug f.eks. de foreslæde links under "Supplerende noter" i semesterplanen.
- Giv herefter en vurdering af, i hvor høj grad dette kan benyttes i jeres ugeopgaveprojekt. Hvilke elementer fra analysen vil I anvende?

Opgave

Lektion 4.3

Udarbejd en liste over projektets risici. For hver skal følgende vurderes:

- Hvor stor er sandsynligheden for, at det sker?
 - Angiv en værdi mellem 0 (det sker slet ikke) og 1 (det sker helt sikkert)
- Hvis det sker, hvor stor vil skaden så være?
 - Angiv en værdi mellem 0 (det betyder intet) og 10 (det er katastrofalt)

Kig på det fra to perspektiver:

- Risici i forbindelse med udvikling af det "virkelige system"
- Risici i forbindelse med jeres projektarbejde.

Opgave

Lektion 4.4

Arbejd videre med opgave 4.3. Vurdér først hvordan I vil identificere de "værst tænkelige" risici. Når disse er identificeret, skal I for hver af dem beskrive:

- Hvordan de kan forebygges – dvs. hvordan sandsynlighed og skadevirkning kan mindskes
- Hvad I vil gøre, hvis de indtræffer

Opgave

Lektion 4.5

Jeres ugeopgaveprojekt skal naturligvis have "høj kvalitet" ;-)

- Undersøg først hvordan man kan identificere kvalitetskrav
- Nævn herefter eksempler på krav, der vil være ønskelige for jeres projekt
- Hvordan vil I sikre at disse krav opfyldes?

Opgave

Lektion 4.6

Der skal udarbejdes et nавигeringsdiagram til ugeopgaveprojektet.

Rapportskrivning

Opgave

Udarbejd et forslag til indholdsfortegnelse (med stikord til hvert afsnit) til jeres ugeopgaveprojekt.

Overvej herefter følgende omhyggeligt:

- Hvad mangler der i forslaget?
- Er emnernes rækkefølge passende?
- Hvad er relevant at vedlægge som bilag?

Analyse

Domænemodeller
Kp. 9

Opgave

Lektion 5.1

Der skal udføres en navneordsanalyse.

- Tag udgangspunkt i en use case og andre af jeres beskrivelser
- Find alle navneord og skriv dem på hver sin seddel
- Vurdér herefter hvilke navneord der er kandidater til klasser
- og hvilke der sikkert er attributter

Opgave

Lektion 5.2

Arbejd videre med løsningen fra opgave 5.1.

- Find associeringer mellem klasser
- Tegn domænemodel

Opgave

Lektion 5.3

Der skal arbejdes videre med løsningen fra opgave 5.2.

- Undersøg, om der er behov for associeringsklasser
- Undersøg, om der er behov for beskrivelsesklasser
- Angiv relevante attributter

Opgave

Lektion 5.4

Tegn et rigt billede – enten af hele jeres system eller over den use case, I valgte at arbejde med i opgave 5.1.

Lektion 5.5

Vælg en klasse fra jeres domænemodel og udarbejd et CRC-kort. Kontrollér dernæst om de relationer, I kommer frem til i CRC –kortet, harmonerer med jeres domænemodel.

Analyse

Aktivitetsdiagrammer

Kp. 28

Opgave

Lektion 6.1

- Vælg en af jeres use cases og udarbejd et aktivitetsdiagram. Angiv evt. pre- og postconditions.
- Vælg dernæst en business proces og udarbejd et aktivitetsdiagram. Det skal opdeles i partitions.
- Udvid en af løsningerne og tilføj objekt flow.

Noget om kvalitet

Opgave

Lektion 6 - kvalitetskriterier

Tag udgangspunkt i jeres ugeopgaveprojekt samt løsning til opgave 4.5 og undersøg:

- Hvilke kvalitetskriterier skal være opfyldt for, at jeres ugeopgaveprojekt (og/eller 3-ugers projektet i januar) får høj kvalitet? Udarbejd en prioriteret liste og begrund de enkelte punkter.
- Hvordan vil I sikre, at disse kriterier opfyldes?
- Hvordan vil I kontrollere, at de bliver opnået?

Analyse og design

Systemsekvensdiagrammer

Pakkediagrammer

Sekvensdiagrammer

Kp. 10 & 12-15.4

Opgaver

Lektion 7.1

Vælg en central use case og udarbejd systemsekvensdiagrammer for et eller flere scenarier.

Lektion 7.2

Vælg et sammenhængende workflow og tegn et systemsekvensdiagram.

Vink: Hvis jeres system er et hotelreservationssystem, kan det f.eks. være alle vigtige handlinger fra en reservation foretages og til opholdet er slut.

Opgave

Lektion 7.3

Der skal udarbejdes et forslag til pakkediagrammer til jeres ugeopgaveprojekt.

- Placér domæneklasser i de relevante pakker.

Opgaver

Lektion 7.4

Tag udgangspunkt i en use case og dens domænemodel. Udarbejd flere sekvensdiagrammer. Sørg for at vælge eksempler, så flest mulig notationer afprøves.

Lektion 7.5

Tag udgangspunkt i et af jeres CRC-kort (Class, Responsibilities and Collaborators) og undersøg, om der er overensstemmelse mellem collaborators og jeres klasser (og operationer) i de tilsvarende sekvensdiagrammer.

Design

**Designklassediagrammer
Kp. 16**

Opgaver

Lektion 8.1

Tag udgangspunkt i domænemodellen fra jeres ugeopgaveprojekt.

- Udvælg de klasser, der skal være i designklassediagrammet.
- Tilføj relevante attributter.
- Brug herefter sekvensdiagrammerne til at tilføje metoder i klasserne.

Lektion 8.2

Undersøg herefter, om alle ansvarsområder (fra CRC-kortene) bliver opfyldt.

Opgave

Lektion 8.3

Fortsæt med eksemplet fra opgave 8.1.

- Tegn aggregerings- og kompositionsstrukturer mellem relevante klasser.

Opgave

Lektion 8.4

Fortsæt med eksemplet fra opgave 8.3.

- Tegn associeringsstrukturer mellem relevante klasser.

Opgave

Lektion 8.5

Fortsæt med eksemplet fra opgave 8.4.

- Tegn generaliseringsstrukturer mellem relevante klasser.

Opgaver

Lektion 8.6

Fortsæt med eksemplet fra opgave 8.5. Undersøg, om kodekvaliteten er bedst mulig.

Lektion 8.7

Tag udgangspunkt i jeres pakkediagram.

- Overvej om der i designklassediagrammet er tilføjet klasser, der hører til i andre pakker end domænelaget.
- Er der opstået behov for et anderledes struktureret pakkediagram?

Design

GRASP

(General Responsibility Assignment Software Patterns)

Design objekter med ansvar

Kp. 17

Opgave

Lektion 9.1

Der skal arbejdes videre med jeres designklassediagram til ugeopgaveprojektet.

- Identificér jeres Creator-, Information Expert- og Controller-klasser.
Der skal måske tilføjes nye klasser.
- Overvej om diagrammet kan ændres, så flere mønstre kommer i anvendelse.

Opgave

Lektion 9.2

Der skal arbejdes videre med jeres designklassediagram til ugeopgaveprojektet.

- Diskutér først hvordan kobling og binding kan relateres til de andre mønstre: Creator, Information Expert og Controller.
- Undersøg om kobling kan gøres mindre og binding større i jeres diagram.
- Designklassediagrammet skal sikkert ændres.

Opgave

Lektion 9.3

I kapitel 18 illustreres det, hvordan et lille udviklingsforløb (use case realisering) kan gennemføres af en gruppe udviklere – dvs. fra use case til kode. Kapitlet omhandler de to systemer, som Larman benytter som eksempler.

- Tag først udgangspunkt i de viste eksempler med NextGen-systemet og gennemgå de viste artifacts med hinanden.
- Overvej herefter om den viste fremgangsmåde er noget I (som nye systemudviklere) kan bruge i jeres fortsatte arbejde med OOAD – i givet fald hvad og hvad-ikke – samt hvorfor.
- Herefter udføres de samme opgaver for Monopoly-eksemplet.

Noget om estimering

Opgave

Lektion 6 - estimering

- Overvej på hvilken måde I vil estimere tids- og ressourceforbruget i 3-ugers projektpériode i januar måned.

Design

Synlighed

Fra design til kode

Relationer mellem use cases

Generalisering/arv

Kp. 19-20 & 30-31.8

Opgave

Lektion 10.1

Tag udgangspunkt i det designklassediagram, I har udviklet til jeres ugeopgaveprojekt.

- Identificér hvilke synlighedstyper der er repræsenteret hvor.

Opgave

Lektion 10.2

Tag udgangspunkt i det designklassediagram, I har udviklet til jeres ugeopgaveprojekt.

- Angiv i hvilken rækkefølge I vil anbefale, at klasserne implementeres. Husk at argumentere for jeres valg.

Opgave

Lektion 10.3

Tag udgangspunkt i jeres use case diagram og undersøg, om der med fordel kan anvendes **include**.

Opgaver

Lektion 10.4

Tag udgangspunkt i jeres designklassediagram og undersøg, om det er relevant at tilføje generaliseringsklasser.

Lektion 10.5

Undersøg herefter, om det er relevant at anvende polymorfi.

Opgave

Lektion 10.6

Tag udgangspunkt i jeres use case diagram og undersøg, om det er relevant at tilføje aktør-generaliseringer.

Analyse og design

**Kvalitet og reviews
Projektplanlægning**

Opgave

Lektion 11.1

Jeres gruppe skal snart deltage i 2 review'es: Jeres eget arbejde skal review'es, og I skal review'e en anden gruppens arbejde.

- Udarbejd en agenda, der kan bruges til et review.
- Udarbejd ligeledes en skabelon til en review-rapport.

Opgave

Lektion 11.2

Udarbejd en oversigt over de diagrammer, beskrivelser og eventuelt programkode, som i ønsker at få review'et i næste uge - til:

- Ugeopgaveprojektet og
- CDIO-opgaven.

Opgaver

Lektion 11.3

- Diskutér fordele og ulemper ved at benytte UP. Sammenlign evt. med en af de andre procesmodeller.
- Udarbejd en faseplan for jeres januar-projekt.

Uge11.4

- Overvej, om der er principper fra XP/SCRUM, som I med fordel kan anvende i jeres projektarbejde i 3-ugers perioden. Begrund jeres valg.

Noget om risici

Opgave

Opgave – risici

- Tag udgangspunkt i jeres ugeopgaveprojekt samt løsningerne til opgave 4.3 og 4.4. Undersøg om besvarelserne kan gøres mere detaljerede.

Opgave vedr. et 3-ugers projekt – til både ITØ'ere og SWT'ere ;-)

- Foretag en risikovurdering i forbindelse med et planlagt koncentreret 3-ugers programmeringsprojekt, der skal udføres af studerende. Tag udgangspunkt i Boehm's top-10 liste og vælg punkter, som kan være relevante for dette projekt.