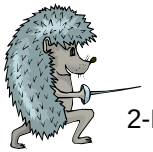
UNIVERSITY
OF OSLO

INF-5150 2005 Oracle

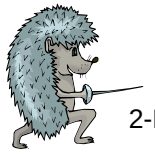
Version 051202

INF 5150



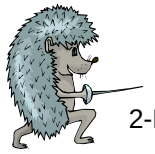
Sven-Jørgen Karlsen

- Her er mine spm. til oppsummeringsforelesningen på fredag: (forkortelser: SD = sekvensdiagram, SM = state machine)
  - 1. Modellsjekking generelt, SD vs. SM spesielt, særlig eksempler og forklaring av sekvensfeil. Hvordan sikrer man seg at SM-ene ivaretar rekkefølgen fra SD-ene, uten å påtvinge designet mindre samtidighet (eng: concurrency)? (Er dette alltid rett fram?)
  - 2. Raffinering:
    - 1. Oppsummering av de avanserte formene for raffinering (utover egenskapsraffinering).
    - 2. Argumentasjon for raffinering fra SD-spek. til SN-design: Ett mønster for argumentasjonen og ett passe klart og konsist eksempel. Jeg synes dette ble litt dårlig belyst på evalueringen av dropp to. Dessuten er løsningsforslaget fra assistene altfor omfattende og uegnet til en tre timers eksamen, med masse detaljer fra SM-implementasjonen.
  - 3. Sikkerhetsanalyse: Oppsummering. Særlig hva som kjennetegner en dårlig analyse, "antipatterns" å unngå?
  - 4. Dialektisk SW-utvikling: Oppsummering.
- Til slutt noen mindre eksamensrelevante spørsmål, hvis det blir tid til overs:
  - 1. SD-syntaks:
    - 1. Hva er "global" og "extra-global" combined fragments og interaction uses? Global, i forhold til hva?
    - 2. Detailing (2. forelesning om SD-er, s. 23, brukes også i forbindelse med grensesnittraffinering, tror jeg): Jeg skjønner ikke helt notasjonen med små SD-er ved siden av eller overlappende til hoveddiagrammet ("NotesTranslator"-diagrammet). Hva er denne notasjonens forhold til de to formene for interaction uses?
  - 2. Å finne mengden av historier fra ett SD: Algoritme eller oppskrift, fra f.eks. bruk i programvare for modellsjekking. Dette er tuklete å utføre riktig manuelt, selv for små eksempler fra forelesningene.
  - 3. Praktisk bruk av raffineringsformalismene: Vi har ført argumenter for at spesifikasjoner og design er egenskapsraffineringer i obligene, men hva kan disse utsagnene brukes til?



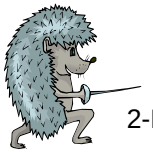
Kjersti og Torunn

- Hei! Vi har følgende tema vi ønsker repitisjon av på forelesning fredag:
- - lage FTA med konkrete eks. fra risk analyse i Drop 2
- - syntaks i StateMachine (SM), hvordan angi transition når vi ikke bruker verkstøy
- - eks. på detailing (interface refinement)
- - gjennomgå et eks. på sammenligning av stateMachine og SD, også hva som forventes ved partiell simulering
- - gjennomgang av eks. på tracer i SD med pos, neg, alt, loop - hvordan håndtere loop i SD i en SM
- - hva betyr assert?
- - vise refinement (supplemeting, narrowing) ved å tegne SD, begge gangene med utgangspunkt i det opprinnelige
- - refinement, repitisjon av bevisføring for både SD og SM
- - eks. på CORAS-modellering (tilsvarende øvelse i forelesning (18/11))



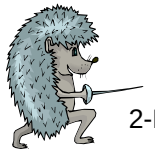
continued

- I tillegg har vi snakket med Atle R. ang. eksamen for 2003. Han kunne ikke hjelpe oss med løsning av oppg. 1a siden det er et krav at systemet skal håndtere vilkårlig mange ordre med en SM. Han mente at den beste måten å løse dette på er å lage logikk som i drop 2 (Controller/Session). Hva er meningen at man skulle ha gjort i oppgaven? Vi diskuterte bruk av submachine, men kom ikke i mål med en slik løsning. Kan vi få svar på dette spm. før fredag?



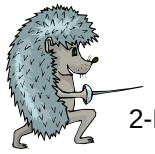
G. Johnsen

- Hei. Hvis det ikke er kommet inn noen forslag til stoff som kan gjennomgås så kunne kanskje en gjennomgang av statemachines (composite states) vært noe?
- Jeg savner litt mer detaljerte og store eksempler i pensumsartiklene.
- Ellers så er det jo alltid interessant å repetere hovedpunktene fra raffinering og sikkerhetsanalyse. mvh



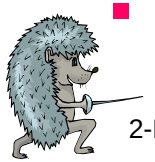
Unni Løland og Astri Ek Larsen

- Hei, Vi har noen spørsmål og tema som vi ønsker at du tar opp på forelesningen i morgen dersom det er mulighet for det.
- - Verifikasjon. Er dette viktig? Vi har hatt om dette på forelesninger, men det har ikke vært en del av obligene våre. Fint om du kan ta en repetisjon av dette.
- - Er feiltrær viktig? Hvordan gjør en beregning av sannsynlighet i feiltrær?
- - Kan du ta et eksempel på bevis av property refinement? - Kan du forklare forskjellen på alt og xalt?
- - På eksamensoppgaven fra 2003 er det en ref i det første sekvensdiagrammet til credit_assesment. Den refen dekker bare livslinjen :Handler, men i sekvensdiagrammet credit_assesment er også livlinjen :CreditCompany med, må ikke da :CreditCompany også være med i sekvensdiagrammet SMSorder?

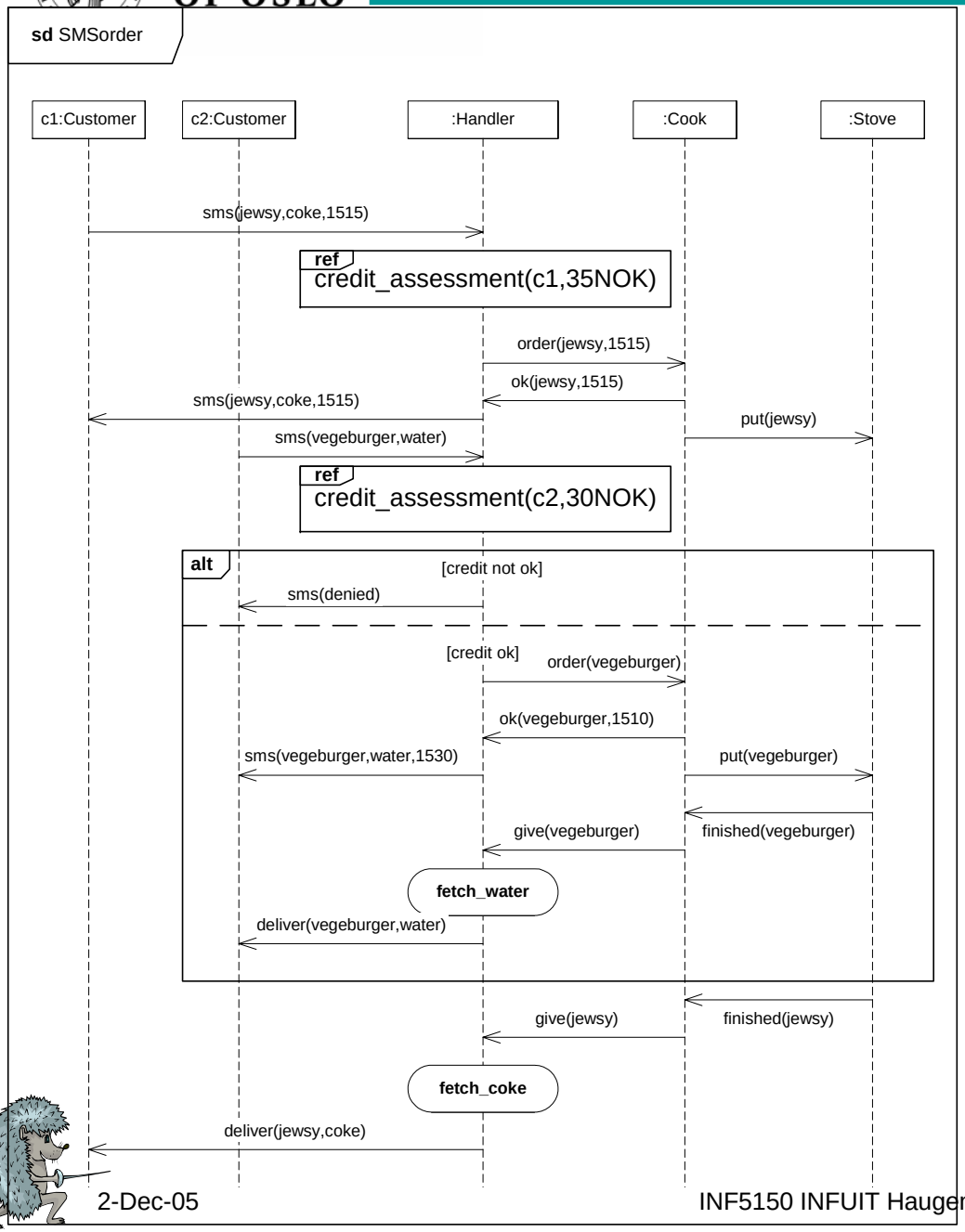


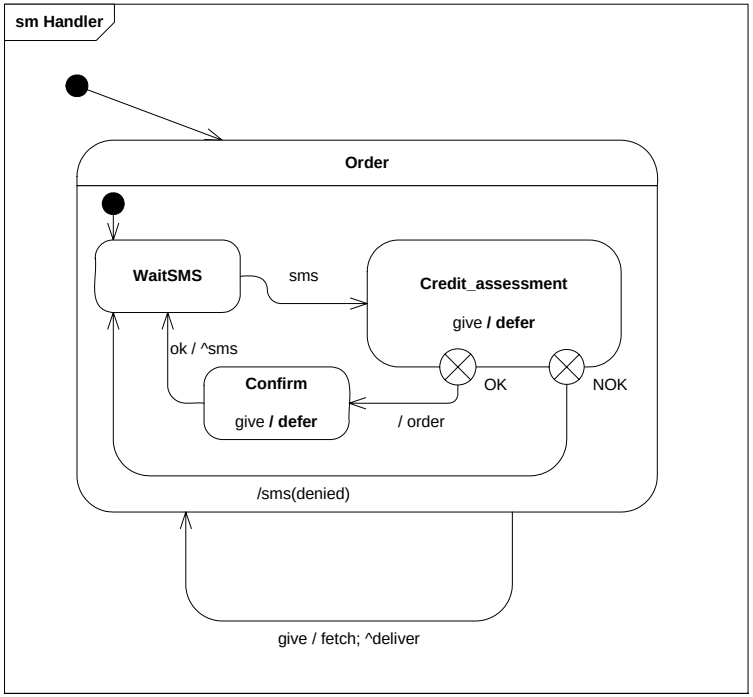
Vibeke

- Tema jeg gjerne skulle ha gjennomgått:
- -bevisføring/refinement for:
  inconclusive tracer
  negative tracer
  nytt eksempel
- -decomposing i tilstandsmaskiner
  bør vi alltid bruke det -selv om det er små systemer
- -risiko
  forklar de forskjellige elementene som inngår
  trusselscenario-gjennomgang
  hva man bør tenke på
  hvorfor er dette nyttig
- -bruk av eventer/tilstand som "WaitingConfirm" inni sekvensdiagrammer
  hva er lovlig
- -forklar composite structure diagram



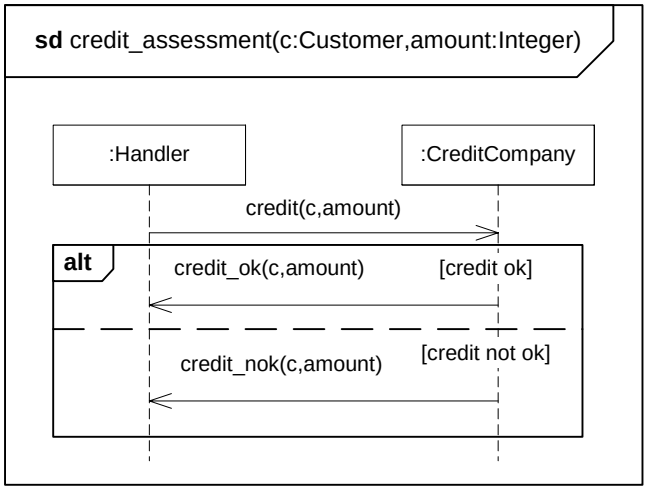
Partial verification – Exam 2003

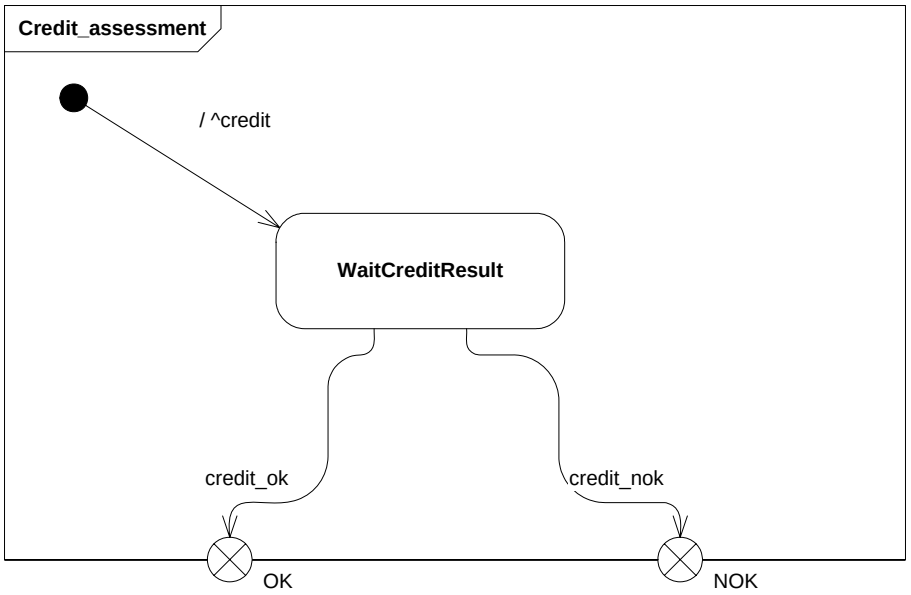




INF 5150

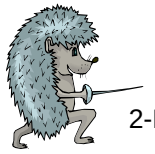
Submachine state (composite state)



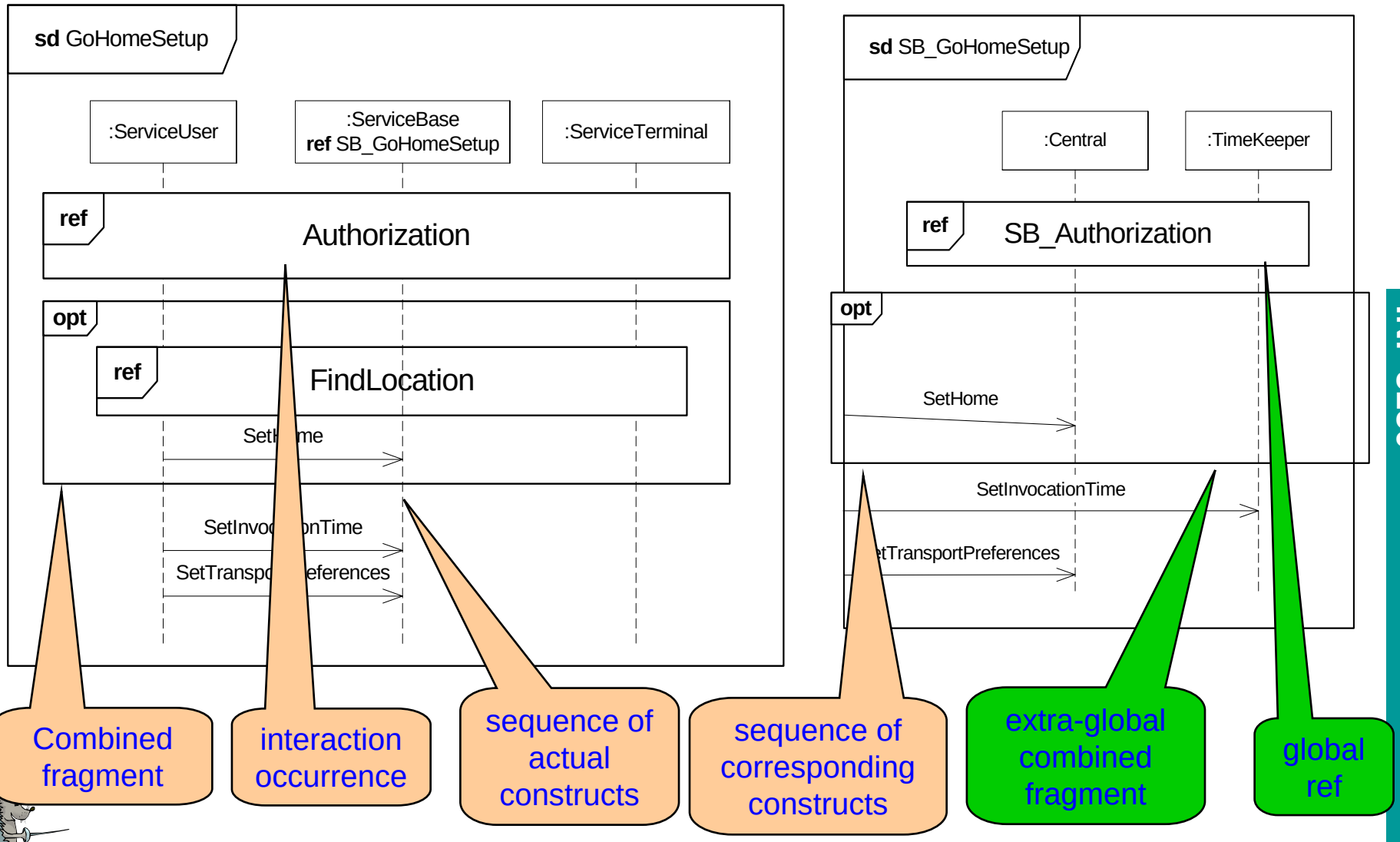


Dialectic Software Development

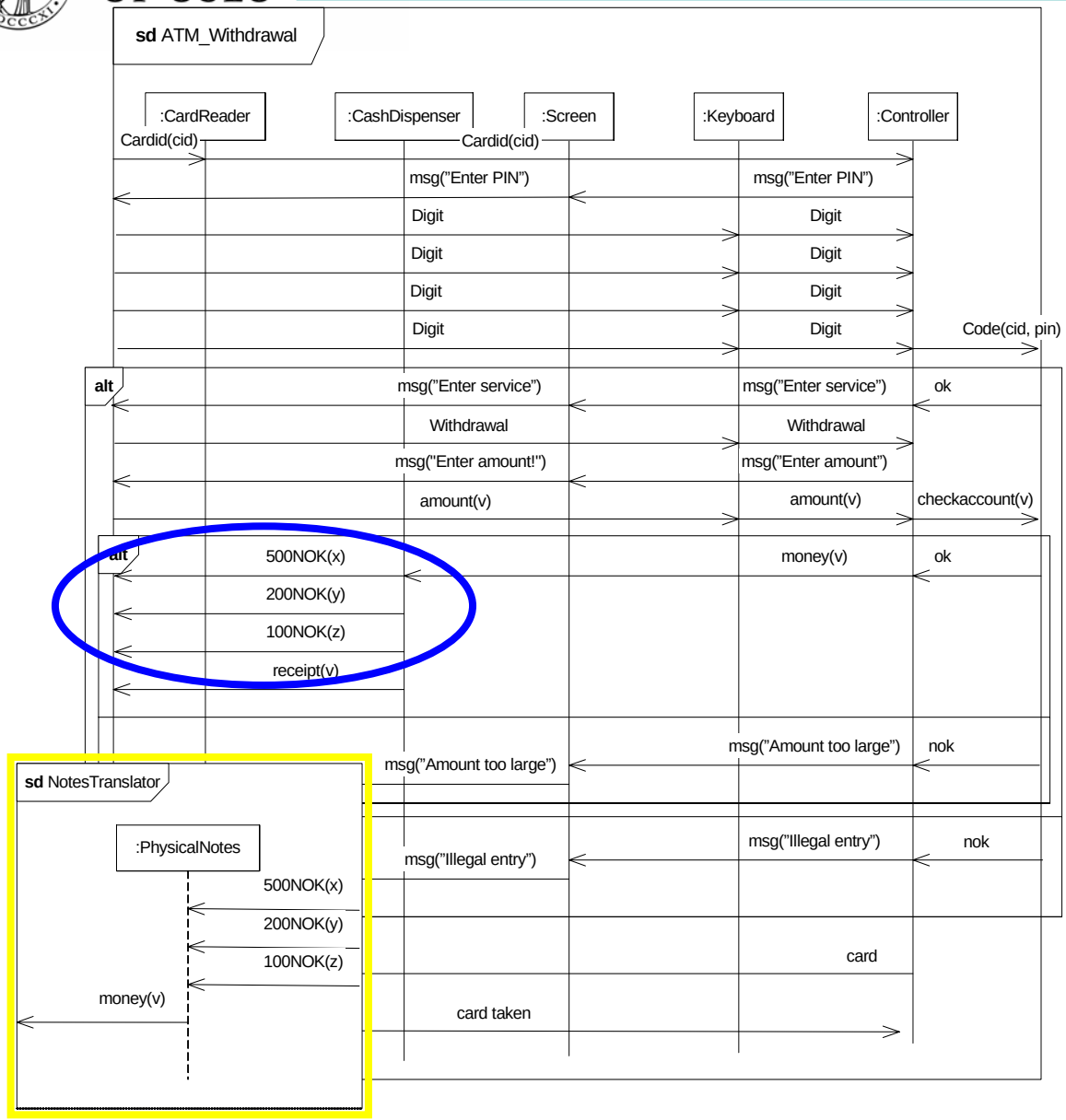
- Software Development is a process of learning
  - once you have totally understood the system you are building, it is done
- Learning is best achieved through conflict, not harmony
  - discussions reveal problematic points
  - silence hides critical errors
- By applying different perspectives to the system to be designed
  - inconsistencies may appear
  - and they must be harmonized
- Inconsistencies are not always errors!
  - difference of opinion
  - difference of understanding
  - misunderstanding each other
  - a result of partial knowledge
- Reliable systems are those that have already met challenges

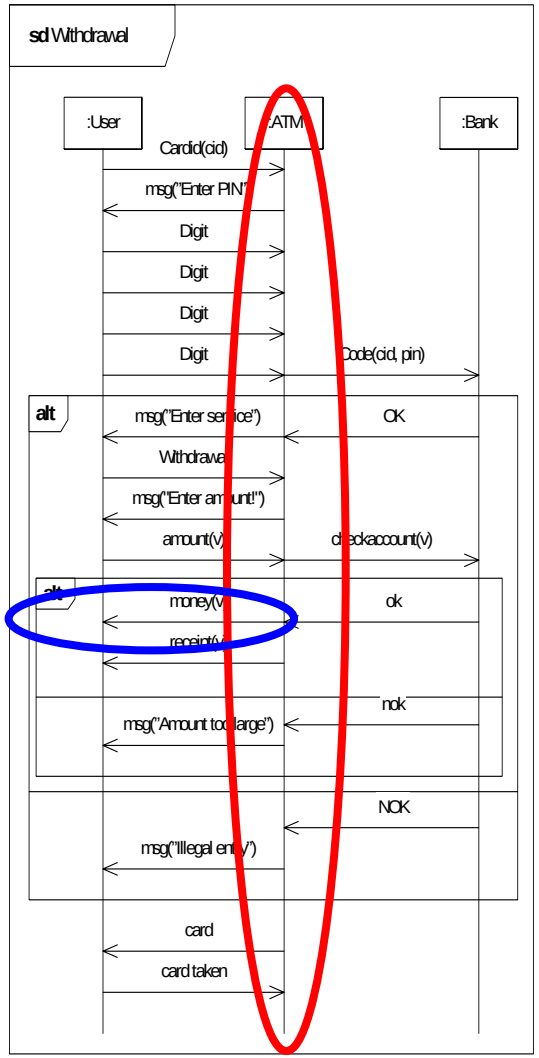


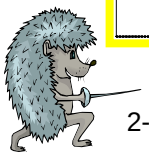
Decomposing covering ref's and combined fragments



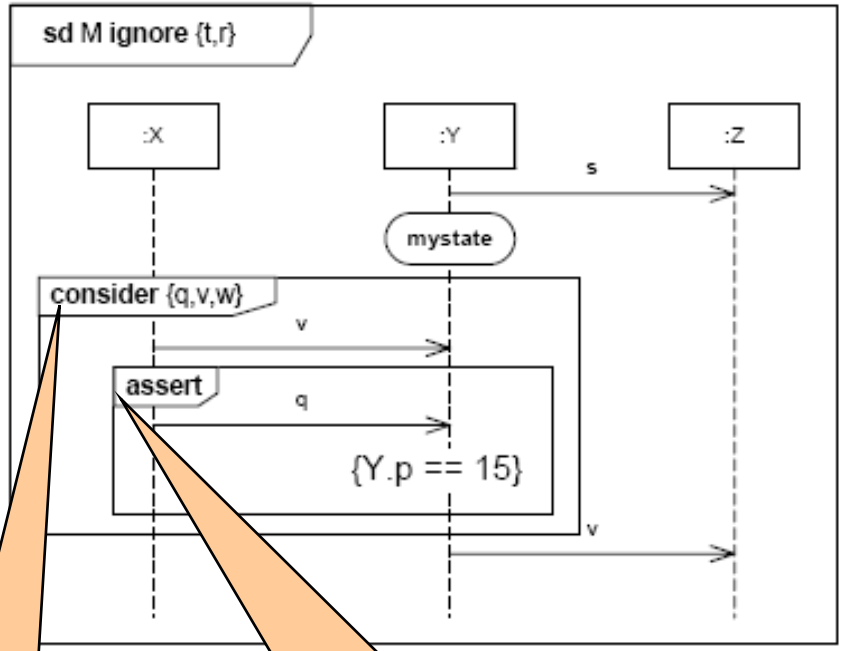
Detailing


sd ATM_Withdrawal
:CardReader
:CashDispenser
:Screen
:Keyboard
:Controller
Cardid(cid)
Cardid(cid)
msg("Enter PIN")
msg("Enter PIN")
Digit
Digit
Digit
Digit
Digit
Digit
Digit
Digit
Digit
Code(cid, pin)
alt
msg("Enter service")
msg("Enter service")
ok
Withdrawal
Withdrawal
msg("Enter amount!")
msg("Enter amount")
amount(v)
amount(v)
checkaccount(v)
alt
500NOK(x)
200NOK(y)
100NOK(z)
receipt(v)
money(v)
ok
msg("Amount too large")
msg("Amount too large")
nok
msg("Illegal entry")
msg("Illegal entry")
nok
card
card taken
sd NotesTranslator
:PhysicalNotes
500NOK(x)
200NOK(y)
100NOK(z)
money(v)


sd Withdrawal
:User
:ATM
:Bank
Cardid(cid)
msg("Enter PIN")
Digit
Digit
Digit
Digit
Code(cid, pin)
alt
msg("Enter service")
OK
Withdrawal
msg("Enter amount!")
amount(v)
checkaccount(v)
money(v)
ok
nok
msg("Amount too large")
NOK
msg("Illegal entry")
card
card taken



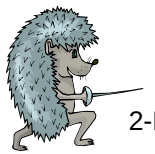
Assert (illustration and text from UML 2.0)



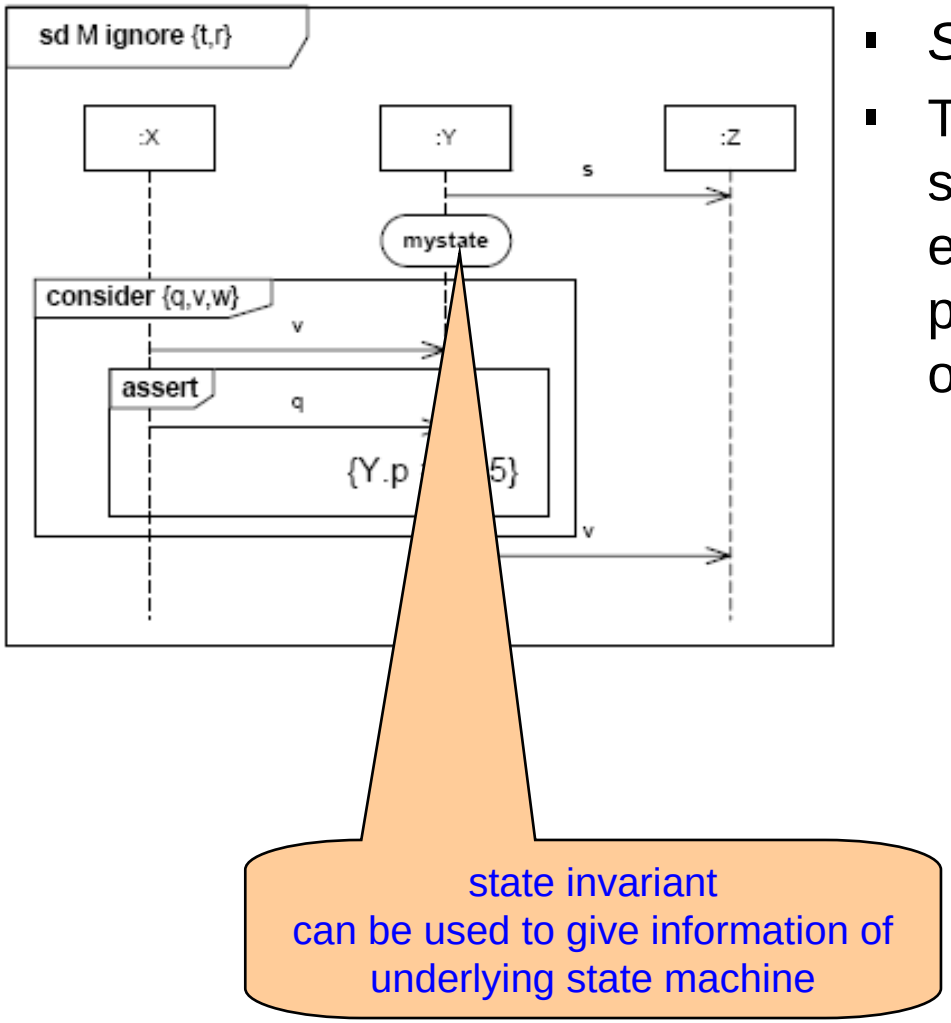
- *Assertion*
- The interactionOperator **assert** designates that the CombinedFragment represents an assertion. The sequences of the operand of the assertion are the only valid continuations. All other continuations result in an invalid trace.
- Assertions are often combined with Ignore or Consider as shown in Figure 14.24. *(given here on this slide)*

gives the vocabulary – not in INF5150

assert
means that what is described in the operand gives positive traces; all other traces are negative



State Invariant (from UML 2.0)



- *State Invariant*
- The State invariant given as a state “mystate” will be evaluated at runtime directly prior to whatever event occurs on Y after “mystate.”

Composite Structure (the JavaFrame way)