# INF5510 - Distribuerte Objekter
# Muntlig eksamen juni 2018

Eric Jul
Institutt for informatikk
Universitetet i Oslo

May 27, 2018

Ved eksamen trekker eksaminanden et av spørsmålene nedenfor, når eksamen starter. Eksamenstiden er ca 25 min, hvorav den første delen skal være en fremleggelse av eksaminanden. Det er tillat å medbringe et A4-ark for hvert spørsmål. - Det er tillat å medbringe alt materiale fra pensumlisten. Det er tillat å bruke eksempler, figurer og annet materiale fra pensum under eksamen.

Det vil også være spørsmål knyttet til Oblig 3 / HE1 og Oblig 4 / HE2, så vær sikker på at har lest opp på løsningen og for å ta med rapportene til eksamen. Den endelige karakteren vil være basert mer eller mindre jevnt på 1. Din presentasjon og svar til spørsmål angående spørsmålet du drar, og på 2. Oblig 3 / HE1 og 3. Oblig 4 / HE2.

At the exam, each student will draw one of the questions listed below when the exam starts. Each exam will take about 25 minutes. Expect to give a short presentation initially. You may bring ONE sheet of A4 sized paper for each question. You may bring any and all material mentioned in the PENSUM (the curriculum listed below). You may use any example, figure or other material from the PENSUM during the exam. There will be questions related to Oblig 3/HE1 and Oblig 4/HE2, so be sure to have read up on your solution and to bring your reports to the exam. The final grade will be based more or less evenly on 1. your presentation and answers to questions concerning the question that you drew, and on 2. Oblig 3/HE1 and 3. Oblig 4/HE2.

## Pensum/Curriculum

The pensum/curriculum appears on the course web pages—and is listed below:

- Erics Ph.D. afhandling
- Niels Chr. Juuls Ph.D. afhandling
- Norm Hutchinsons Ph.D. afhandling
- 1992 ISMM artikklen om Distributed Garbage Collection
- 1991 SP&E artikklen om Emerald
- Emerald Language Report
- IEEE Software Engineering artikklen
- TOCS Emerald artikklen
- 2007 HOPL Emerald artikkel

## Spørsmål

1. Gør rede for Emeralds sprogmekanismer som understøttet mobility - herunder call-by-move.
2. Gør rede for Emeralds måde at lave nye objekter på herunder object constructors, immutable objects og hvorfor og hvordan man laver klasser i Emerald inklusive hvordan man laver subklasser og objekter af subklasser. Forklar parametre til klasser. Forklar hvorfor en klasse kan brukes som en type.
3. Gør rede for Emeralds typesystem - specielt conformance, herunder hvorfor det er vigtigt i et distribueret system. Forklar pizza/junk eksemplet fra SP&E artiklen. Forklar NIL, NOONE og ANY og hvordan de passer ind i typesystemet. Forklar view-as og restrict-to, gjerne via eksempel.
4. Gør rede for remote invocations i Emerald og for hvordan de er implementeret i det oprindelige Emerald. Forklar samspillet med mobility, call-by-move, call-by-visit og forklar break-even for at flytte et parameterobjekt med ifm et kald.
5. Gør rede for hvordan Emeralds comprehensive, robust, distributed garbage collector virker.
6. Gør rede for immutability og hvorfor det er vigtigt i et distribueret system.
7. Gør rede for hvordan object mobility er implementeret i det oprindelige Emerald. Brug gjerne et SIMPELT eksempel.
8. Gør rede for Emeralds attach begreb, herunder motivation for det, hvordan det bruges og hvordan det er implementeret i det oprindelige Emerald.

## Questions in English

1. Explain Emerald's language mechanisms that support mobility - including call-by-move.
2. Explain how to create new objects in Emerald including the concept of object constructors, immutable objects and how and why to make classes in Emerald including how to make subclasses and objects of such subclasses. Explain class parameters. Explain why a class can be used as a type.
3. Explain Emerald's type system - specifically conformance and why it is important in a distributed system. Explain the pizza/junk example from the SP&E article. Explain NIL, NOONE and ANY and how they fit into the type system. Explain view-as and restrict-to; use examples, if possible.
4. Explain remote invocations in Emerald and how they are implemented in the original Emerald. Explain the interaction between the concepts of mobility, call-by-move, call-by-visit and explain break-even when moving a parameter object in connection with an invocation.
5. Explain how Emerald's comprehensive, robust, distributed garbage collector works.
6. Explain immutability and why it is important in an distributed system.
7. Explain how object mobility is implemented in the original Emerald. Use a SIMPLE example.
8. Explain Emerald's concept of attachment, including the motivation for it, how it is used and how it is implemented in the original Emerald.