IN5570/IN9570 - Distribuerte Objekter
Muntlig eksamen juni 2023
Oral Exam June 2023

Eric Jul
Institutt for Informatikk
Universitetet i Oslo

May 4, 2023

Ved eksamen trekker eksaminanden et av spørsmålene nedenfor, når eksaminandens eksamination starter. Eksamenstiden er ca 15-20 min, hvorav den første delen skal være en fremleggelse av eksaminanden (cirka 5 minutter plus/minus nogle minutter). HVIS du trekker et spørsmål, du ikke liker, så gjør fremleggelsen kort og bed om spørsmål. 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. Du kan tale dansk/norsk bokmål/svensk/engelsk – og du kan velge om eksaminator skal bruke engelsk eller dansk – sensor taler norsk og engelsk og forstår svensk og dansk. Examen teller 1/2 av karakteren og de to home exams teller hver 1/4. Karakteren gis når alle er eksaminert.

At the exam, each student will draw one of the questions listed below when the exam starts. Each exam will take about 15 minutes. Expect to give a short (say 5 minutes) presentation initially. You may speak Norwegian Bokmål/Swedish/Danish/English at your discretion. You can ask the examiner to speak either Danish or English. The censor speaks Norwegian and English and understands Swedish and Danish. 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.

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
- 1988 TOCS Emerald artikklen om Object Mobility
- 2007 HOPL Emerald artikkel
- Tony Hoare: 1974 artikkel om Monitors
- Gordon Moores 1965 artikkel
- Lysets hastighed / Speed of light
- Lecture Materials published on the course web site

Spørsmål

1. Gør rede for Emeralds sprogmekanismer som understøttet mobility – e.g., locate, call-by-move, node, ...

2. Gør rede for Emeralds måde at lave nye objekter på herunder object constructors. Forklar immutable objects og hvorfor de er vigtige i Emerald. Forklar hvorfor der IKKE er klasser i Emerald. Forklar hvordan Emerald HAR klasser via syntactic sugaring. Forklar 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 type system – specielt conformance, herunder hvorfor det er vigtigt i et distribueret system. Forklar pizza/junk eksemplet fra SP&E artiklen gjerne grundigt. Forklar lattice of types, NIL, NOONE og ANY og hvordan de passer ind i typesystemet. Forklar view-as og restrict-to, gjerne via eksempel.

4. Gør rede for hvordan Emeralds comprehensive, robust, distributed garbage collector virker. Specielt mark-and-sweep, og hvordan Emerald bruker en read-barrier. Det er viktig at du kan forklare algoritmen som brukes til at avslutte Distributed GC.

5. Gør rede for immutability og hvorfor det er vigtigt i et distribueret system. Forklar hvorfor immutability kan forbedre performance. Brug gjerne simple eksempler.

6. Gør rede for hvilke principper, der gjør, at Emerald er effektivt implementeret og hvordan object mobility er implementeret i det oprindelige Emerald. Brug gjerne simple eksempeler.

7. Gør rede for Emeralds attachment begreb, herunder motivation for det, hvordan det bruges og hvordan det er implementeret i det oprindelige Emerald. Bruk gjerne eksempler.

Questions in English

1. Explain Emerald's language mechanisms that support mobility – including locate, call-by-move, node ...

2. Explain how to create new objects in Emerald including the concept of object constructors. Explain immutable objects and why they are important in Emerald. Explain why there is no class concept in Emerald and how you write an object that behaves like a class. Explain how Emerald HAS classes via syntactical sugaring. Explain how to make subclasses and objects of such subclasses. Explain class parameters. Explain why a class can be used as a type. It can be beneficial to show brief, illustrative examples.

3. Explain Emerald's type system - specifically conformance and why it is important in a distributed system. Explain the lattice of types and NIL, NOONE and ANY and how they fit into the type system. Explain view-as and restrict-to; use examples, if possible. Explain the pizza/junk example from the SP&E article carefully.

4. Explain how Emerald's comprehensive, robust, distributed garbage collector works. Explain basic mark-and-sweep – and how Emerald uses a read-barrier. It is important that you can explain the termination algorithm.

5. Explain immutability and why it is important in an distributed system. Explain how immutability can improve perfromance. You are encouraged to use examples.

6. Explain the principles that made Emerald effective and efficient concerning performance in the original implementation and how object mobility is implemented in the original Emerald. Using simple examples is encouraged.

7. Explain Emerald's concept of attachment, including the motivation for it, how it is used and how it is implemented in the original Emerald. You are encouraged to use examples.