Master thesis topics: Programming & Software Engineering

Plan

- Assoc. Professor II Eyvind Axelsen
- Professor II Ketil Stølen
- Dr. Parastoo Mohagheghi, NAV
- Assoc. Professor Arne Maus
- Lecturer Yngve Lindsjørn

- Professor Eric Jul
- Assoc. Professor Viktoria Stray and Jørgen Hesselberg. Start-up lab
- Assoc. Professor Dino Karabeg
- Assoc. Professor II Arne Jørgen Berre
- Assoc. Professor Antonio Martini
- Siri Moe Jensen and Ragnhild Kobro Runde

Master thesis suggestions 2017

Eyvind W. Axelsen (eyvinda@ifi.uio.no)

Associate Professor II at Ifi, UiO
- I.e., I'm a *part time* employee
- Programming and Software Engineering (PSE) research group

Head of Software Development at Fürst Medical Laboratory
- Biggest laboratory in Norway, > 400 employees
- > 10 000 patients per day, > 100 000 analysis results per day
- Strategic focus on IT

Some ideas for theses follow. Do you have your own idea? Get in touch!

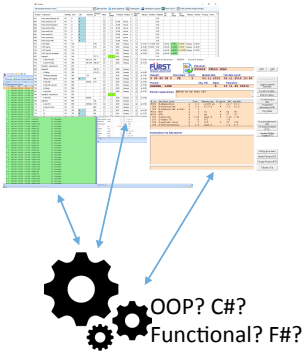
DSL for decision making based on analysis results

- DSL = Domain Specific Language
  - A programming language that is *specific* to the problem at hand
  - A programming language that YOU make!
- Lots of hard-coded rules today
  - Results from single or multiple analyses, patient's history and results from genetic tests, threshold values, etc → decisions about further analyses, comments, warnings, etc
  - Discrepancy between the understanding of the medical profession and the software engineers – are the rules correct?
- Make a language that "everyone" can understand, that is transparent
  - Users without knowledge of informatics
  - But with strong medical skillset
  - How can we make something that is both understandable and sufficiently expressive?
  - Truth tables, programming or modeling languages?
- Requires:
  - Solid programming skills (Java/C#/similar)
  - Some compiler knowledge
  - Interest in understanding the medical domain
- Recommended subjects: INF3110 and INF5110

People:
Eyvind W. Axelsen, Ifi/Fürst, supervisor
Trygve Tjade, Fürst, chief attending physician («overlege»)
Trond Ranheim, Fürst, chief attending physician («overlege»)
Contact: eyvinda@ifi.uio.no

Short(?) master: reimplement OO services with functional programming on the .NET platform

- Services written in C# for exchange of medical information
  - Message based communication over TCP
  - In-house/external
  - Real-time constraints
- Can these be written in a more concise/safer/better manner with F#?
  - Compare and contrast the languages/paradigms w.r.t. this use case
  - Do an analysis of the benefits (if any) or downsides (if any) to each implementation in a real-world scenario
- Requires:
  - Solid programming skills (preferably Java/C# and ML/Haskell/F# or similar)
  - Interest in programming languages
- Recommended subjects: INF3110, maybe INF2810?



People:
Eyvind W. Axelsen, Ifi/Fürst, supervisor
Fürst software team
Contact: eyvinda@ifi.uio.no

Area of interest

- Cyber-security
- Cyber-risk
- I&T

7

Tool for aggregation of cyber-risk

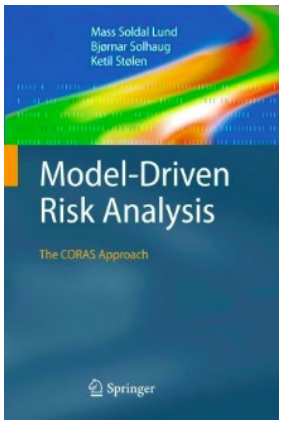
- Automatic aggregation
- Large companies
- Presentation at different organisational levels

- Will involve theory, open-source software development and empirical studies

8

CORAS portal

- Portal for making CORAS comprehensible to beginners
- Build on theory
- Try to exploit modern technology
- Make decisions based existing theory as much as possible
- Test out solution in empirical studies



9

There are many possibilities at SINTEF

- SINTEF will have a separate session for presentation of possible MSc-topics at IFI October 2, 16:00

10

Ta din masteroppgave i det mest spennende IT miljø i Norge

Parastoo.Mohagheghi@nav.no
NAV IT, Sannergata 2, Oslo

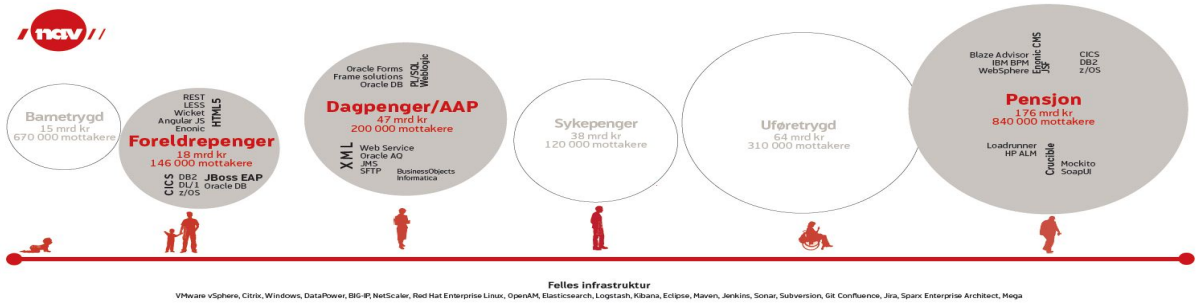
Flere i arbeid

Bedre brukermøter

Økt kompetanse

21.09.2017 / Presentasjon på UiO

NAV (IT) - tall og fakta



Stort volum 2.8 millioner mennesker bruker NAVs tjenester Over 500 000 IT-relaterte prosjekttimer årlig 15 000 kroner utbetales hvert sekund	**Høy kompleksitet** Over 300 datasystemer som samhandler 7 500 fysiske og virtuelle servere Utvikling på flere teknologier og plattformer
Sterk kompetanse Blant Nordens fremste IT-miljøer Over 620 dyktige medarbeidere Internasjonal prisvinner	**Veien videre** Bygge opp intern utviklerkapasitet Større ansvar der jobben gjøres Smidig tilnærming, kontinuerlige leveranser

Målbildet vårt

- Brukertilpassede digitale kanaler
- Automatisert saksbehandling
- Datadrevet intelligent oppfølging
- Balansert sourcing – bedre bruk av markedet
- Smidig utvikling – kontinuerlig leveranse

Utvikling hos oss

- Noe smidig, noe fossefall, på vei mot flere smidige team
- Vi skal utvikle mer selv
- Mange spennende prosjekter som er viktige for samfunnet
  - Sykepenger, foreldrepenger, plattform for arbeidsmarkedet

Problemstillinger for masteroppgaver

- Smidige team
  - Tight-Loose-Tight (Forretningsstyrt, autonome team og sikre gevinster); hva betyr det, hvilke empiri finnes, hvordan realisere det?
- Metrikker på ulike nivå
  - Hvordan måle kost / tid/ kvalitet / omfang i smidige team? Og til hvilke formål?
  - Hvilke metrikker for hvem? utviklere/ produkteiere / virksomhetsstyring
  - Hvilke data kan samles i våre verktøy og hva bør vi kunne samle?
- Smidig i en stor organisasjon
  - Sammenligninger av metoder og anbefalinger
  - Hvordan inngår selvstyrte og ansvarlige team i en større virksomhet?

Hvorfor gjennomføre masteroppgave hos oss?

- Du får sitte sammen med oss
- Du får se på reelle problemstillinger
- Alle tema er relevante for næringslivet! Tenk på din CV!

- Vil du vite mer om utvikling hos oss? Se foredrag fra Javazone 2017

Her kan du se foredraget

Design of Efficient Parallel Algorithms

This thesis will look at different ways, different design patterns, for making parallel algorithms for *k* computing cores on a shared memory computer, a multicore PC.

Assuming that for a given problem, we have at least one good sequential algorithm.

We can :

1. **Divide and combine/map and then reduce at termination (working on shared data)**
2. **Introducing copies of central data (local calculations before combination)**
3. **Repeated map-reduce of type 1 or 2.**
4. **Divide the problem into *k* disjunct subproblems and solve them completely in parallel; each solved with the sequential algorithm. Then combine these solutions in parallel to a solution to the big, original problem**
5. **……**

In the thesis, for a small set of problems with sequential algorithms:

- **Make more than one parallel algorithm for each of a small set of problems,**
- **Implement and test empirically their performance**
- **Try to conclude on more general ways (design patters) for making efficient parallel algorithms.**

Theory, programming and empirical evaluation of algorithms.

Supervisor: Arne Maus (arnem@ifi.uio.no)

Yngve Lindsjørn
ynglin@ifi.uio.no
91549139

Master topics – Teamwork and Large Scale Agile Software Development

1) Large-Scale Agile Software Development. Agile methods were first used in small projects with little criticality. How can agile practices be adapted and combined with traditional practices to function effectively in large-scale development and multi-team environment?
   Challenges:
   - System architecture across teams
   - Working agile in "non-agile" organizations and settings
   - Consistency across teams for the development practices
   - Inter team dependences
   - Team leader role and product owner role
   - Handling requirements in distributed development
2) Conduct a literature review of research on teamwork and the relation to team performance and personal success in software development
3) Conduct an empirical study (qualitative and/or quantitative) on how teamwork factors such as team leadership, team cohesion, communication and self management effect team performance and personal success in software development teams.

Master's Thesis Proposals

Professor Eric Jul
Programming and Software Engineering Group
ericbj@ifi.uio.no

Fall 2017

I teach INF5510 "Distributed Objects" every Spring.
Veiledning: enten på dansk or English (American)
Strong believer in *Learning-by-doing* –so project are much about programming

Project areas:

- Object-oriented Programming Languages
- Distributed Objects and Cloud Programming
- Design Patterns

Detailed presentation on October 4th, 2017 at 16:00 in OJD Meeting Room 10167

Distributed Objects, Cloud, Language Mechanisms

- **Elastic Expansion of Cloud Resources**
  - Dynamically adapting resource in a Cloud Computing Center by combining vertical and horizontal scaling-a simulation. *This project is in cooperation with Bell Labs Ireland, who will fund a student internship for 3-4 months*. So if a stay at a top industrial research lab in Ireland is interesting, consider this project.
- **Implementing the TRACK mechanism in Emerald**
  - TRACK is a proposed language construct that allows an object, A, to keep track of where another object is currently located in a distributed system.
  - Combination of implementation (both compiler and runtime) and language level usage of the concept.
  - Co-author an article with me on the subject for a conference
- **Type-based Access Control, Locks, and Authentication**
  - Using the restrict-to mechanism to limit access control
  - Implement a number of examples of how to provide language based control mechanism
  - Co-author an article with me on the subject for a conference

Distributed Objects, Language Mechanisms

- **Watch dog service**: Write an Emerald program that implements a watch dog service that runs on Planetlab. The service provides the ability to keep a user service running in a number of replicas by monitoring the replicas and starting up new replicas when previous ones disappear, e.g., due to node crashes.
- **Network map**: Write an Emerald program that maps out the internet nodes where your Emerald program runs. It is to replicate itself onto all available nodes and then monitor which nodes are up, how long they have been running, and what the round trip delays are between the nodes.
- **Comparison of the design of Java and Emerald**. Compare the design decisions made for these two languages. Use programs to illustrate strengths and weaknesses of each design decision.

Design Patterns

- **Design Patterns for Mobile Devices**
- **Design Patterns for Cloud Computing**

A comparative study of either area.

Must be combined with the actual implementation of multiple design patterns in a prototype system

- **How languages affect design patterns**

An analysis of how different language designs affect various design patterns, e.g., some languages have features that directly implement some design patterns, such as iterator or singleton.

The analysis must be combined with experimentation, i.e., implementing some design patterns in a number of different languages.

Presentation of projects in detail October 4th, 2017

These projects will be presented by Eric Jul on

October 4th, 2017 at 16:00

In

Meeting Room OJD 10167 (tenth floor)

Continuous Improvement in Agile Companies

- Supervisors: Viktoria Stray and Jørgen Hesselberg
- Comparative agility framework
- A four-step process for making organizational impediments visible, creating cross-functional alignment and identifying actionable steps for improvement.

24

MS Topics in Knowledge Federation

The task of knowledge federation is to make knowledge, information and information technology more useful to humans by developing new ways in which information is created, organized and shared. Think of it as algorithm design or as technological innovation on a very high level, where 'discoveries' are new ways in which science, journalism, education, governance and other basic institutions might operate. Knowledge federation is an exciting new field to work in. It is developing fast, and we need good people.

So if you may be interested in growing with a new field, or in applying your programming and/or other skills toward creating a better world, through better use of knowledge, we can formulate a MS project that suits your talent and interests. Our MS theses may or may not involve programming. Here are some examples.

- **Domain Map** is a tool which a scientific discipline or any other community may use to organize its knowledge. The task is to create a simple prototype.
- **Collaborology** is a university course model, a redesign of the way education operates. By studying and describing this model, and perhaps developing it further, you may learn and explain on a concrete example how information technology can be applied to recreate basic institutions.

Dino Karabeg
0JD 10463
dino@ifi.uio.no
tel. 412 92 049

Master thesis topics
Arne.J.Berre@sintef.no and arneb@ifi.uio.no
(ref. course INF5120)

- Big Data from IoT with AI/Machine Learning– for various Application domains/partners
- Big Data and IoT Technologies for Agriculture, Forestry, Fisheries and Aquaculture (Data Bio project – www.databio.eu)
- Software Engineering for Platform-based system development with IBM BlueMix, Microsoft Azure and Apache Big Data Stack
- Big Data and AI for Energy and Smart Grids (Energytics – Hafslund)
- Machine Learning/AI Analytics applied to forecasting and predictions in various domains (Smart Cities, Fisheries, Energy, ….)
- Big DataTechnologies and IoT for Agriculture and/or Forestry (Data Bio project)
- Industry 4.0 in practice for Norwegian SME Production companies (with SME companies producing windows and doors etc.)
- Big Data and IoT Technologies for Smart Cities (with Oslo kommune)
- IoT and NoSQL databases - Big Data architectures and benchmarking (EU DataBench project from January 2018)
- Thesis topics are related to the GEMINI Centers for Big Data, IoT and AI/Machine Learning with SINTEF, UiO and NTNU

Antonio Martini

- technical debt
- software quality
- economic benefits of Refactoring
- Also:
  - software architecture
  - communication,
  - prioritization,
  - customer feedback

– would primarily like to support students doing studies with companies

Master thesis proposals: Teaching and learning programming

Siri Moe Jensen

(siriamj)

Ragnhild Kobro Runde

(ragnhilk)

Mental models of programming

Background:

- Viable mental models are important when learning to program.
- Different programming languages and paradigms encourage different mental models.

Task:

- What are the main differences between the mental models, and how does this influence the learning in later programming/computer science courses?

Contact: siriamj@ifi.uio.no, ragnhilk@ifi.uio.no

Programming «for all»

Background:

- More and more children/students get *some* programmering experience through school/leisure activities.
- Large growth in access to electronic data from various sources and vast application areas.
- Increased focus on programming/computations in many of the study programmes at the Faculty (mathematics, physics, biology, ...)

Task:

- What are the implications of this for computer science as a discipline and education?
- Basic programming education – what is and should be similar/different depending on the intended usage?

Contact: siriamj@ifi.uio.no, ragnhilk@ifi.uio.no

General themes and principles in computer science

Background:

- Computer science as a discipline includes a number of recurring themes and general principles such as abstraction, complexity, security, and concurrency.
- These have broad application to the field of computer science, and not only relevance to the domains in which they were introduced.

Task:

- Investigate students' understanding of one or more themes/ principles, targeting students at different study levels.
- Can this be assessed using traditional assessment methods?

Contact: siriamj@ifi.uio.no, ragnhilk@ifi.uio.no