Introduction to Writing a Thesis

Tips, Tools, and Recommendations

Hein Meling

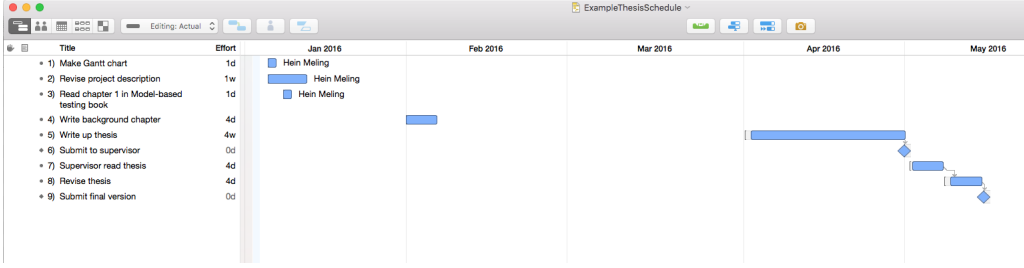
University of
Stavanger
hein.meling@uis.no

January 2016

Planning Your Thesis Work

- Deadlines:
  - Bachelor: May 15
  - Master: June 15
  - Only head of department can grant extensions (he usually won't)
  - Extending due to sick leave: need sykemelding from doctor;
    - usually only if more than a week of illness
- Not a whole lot of time!
- Need careful planning
- Important to stay focused

Gantt Chart


ExampleThesisSchedule
Editing: Actual
Title
Effort
Jan 2016
Feb 2016
Mar 2016
Apr 2016
May 2016
1) Make Gantt chart 1d
2) Revise project description 1w
3) Read chapter 1 in Model-based testing book 1d
4) Write background chapter 4d
5) Write up thesis 4w
6) Submit to supervisor 0d
7) Supervisor read thesis 4d
8) Revise thesis 4d
9) Submit final version 0d
Hein Meling
Hein Meling
Hein Meling

Project Description

- Project descriptions are in some cases a bit vague
- Project proposer does not always know what he/she wants
- Write your own version
- Headings:
  - Background
  - Motivation
  - Objectives
  - Gantt Chart with Tasks and Milestones
- Submit project proposal by **Wednesday January 20**

Project Description

PROJECT ASSIGNMENT

September 17, 2015

Project title	Distributed EDI service providing medical and health administrative messages between actors in the health sector
Student's name	Hans Henrik Grønsleth [hansheg@stud.ntnu.no]
Course	TTM4501 – Telematics, Specialization Project
Department	Department of Telematics
Supervisor	Magne Mæhre, Norsk Helsenett [magne.maehre@nhn.no]
Responsible professor	Bjarne Helvik [bjarne@item.ntnu.no]

Background Norsk Helsenett SF runs a national Electronic Data Interchange (EDI) service that communicates medical and health administrative messages between agents in the health sector. Today the system runs on a centralised, hardware fault tolerant Linux solution. The SMTP and POP3 protocols are used to send and retrieve the (encrypted) messages, respectively.

Motivation Considering the nature of the messages, it is critical that the system is available at all time. To increase the robustness of the system, e.g. handle failure in the communication infrastructure or crashes in data centers, and to facilitate maintenance without suspending the system, it should be possible to send messages to and retrieve messages from multiple servers across the country.

IP Anycast provides an elegant way of communicating with the nearest[1] available server that is transparent to the application layer. Using IP Anycast can therefore reduce the change needed at the end user when changing from a centralized to a distributed solution.

A Proof-of-Concept environment with 3 virtual servers has been established to demonstrate the feasibility of a possible distributed solution. Here, incoming messages at a user's primary server are copied to an outgoing queue and spread to all other servers. Similarly, when a user retrieves messages from his primary server, a revoke message is put in an outgoing queue and sent to all other servers.

Objectives The long term objective is to make the current centralized solution into a more robust distributed solution. The objectives of the project that will be conducted this fall are:

1. Give a description of the state-of-the-art on fault tolerance techniques used in distributed systems with similar characteristics.
2. Suggest a fault tolerant distributed version of the current centralised system that uses IP Anycast, using the Proof-of-Concept as a starting point. This will include addressing detection of inconsistencies, detection of loops, locking problems and scalability. Even though security and performance are essential aspects of the system, these will not be addressed in this project.
3. If time, inject failures in the (possibly modified) Proof-of-Concept environment and investigate the fault handling.

Work Process

- Weekly supervision: 20 minutes
  - Must be efficient; come well-prepared
  - You are expected to drive the meetings
  - What's been done; what are the challenges
  - Action Points for next meeting

Work Process

- Weekly supervision: 20 minutes
  - Must be efficient; come well-prepared
  - You are expected to drive the meetings
  - What's been done; what are the challenges
  - Action Points for next meeting
- After each meeting
  - Write a short summary (max 5 lines)
  - Action Points
  - Send to me (or shared Dropbox folder)

Work Process (2)

- Towards a report
  - Three deliverables
  - Read and comment on each others reports
  - Learn a lot from reading what others write

Work Process (2)

- Towards a report
  - Three deliverables
  - Read and comment on each others reports
  - Learn a lot from reading what others write
- Deliverables (Mondays):
  - February 15
  - March 14
  - April 25

Work Process (3)

- I will read and comment on your report too:
  - Once at the end and
  - Some of the deliverables
- In some cases: review code
- Will help as much as I can by email or Slack

When to Start Writing?

When to Start Writing?

- Today!

When to Start Writing?

- Today!
- You can set up the document

When to Start Writing?

- Today!
- You can set up the document
- Prepare a disposition (fill in the headings)

When to Start Writing?

- Today!
- You can set up the document
- Prepare a disposition (fill in the headings)
- You can always change things later

How to structure the report

- Abstract (300-500 words)
- Introduction (3-5 pages)
- Background, Technology Review, Related Work (10-20 p.)
- Design, Model, Architecture (10-30 p.)
- Implementation (10-20 p.)
  - Technical details of interest, optimizations, algorithms, ...
  - Don't explain every bit of code; only the interesting parts
- Evaluation, Discussion, Insights (10-30 p.)
- Conclusion (300-500 words)

Tips for the report content

- Name chapters appropriately; other names than those on previous slide is ok.
- Start Chapter/Section with a brief summary of what the reader can expect to learn from reading it.
  - Thus far, we have presented Replacement for a single Paxos instance. In this section, we explain how to apply Replacement to an RSM that executes a sequence of Paxos instances.
- Insert paragraph breaks (avoid too long paragraphs)
-

How to Work?

How to Work?

- Switch between coding and writing!

How to Work?

- Switch between coding and writing!
- Expect to rewrite everything you write;
  - So don't spend time making it perfect

How to Work?

- Switch between coding and writing!
- Expect to rewrite everything you write;
  - So don't spend time making it perfect
- Iterate to improve report to make it perfect

How to Work?

- Switch between coding and writing!
- Expect to rewrite everything you write;
  - So don't spend time making it perfect
- Iterate to improve report to make it perfect
- Why work this way?

How to Work?

- Switch between coding and writing!
- Expect to rewrite everything you write;
  - So don't spend time making it perfect
- Iterate to improve report to make it perfect
- Why work this way?
  - Two modes of thinking
  - Makes you think differently about the problem

Specification: Thinking Before Coding

- Write specifications:
  - high-level specification of system
  - low-level specification for each function
- Textual specifications are fine!
- Also: Express ideas in diagrams and figures
- Simplicity

Modeling Languages

- Unified Modeling Language (UML)
  - Use Case Diagrams
  - **Class Diagrams** (Entity Relation)
  - **Sequence Charts**
  - **State Machine Diagrams**
- **Flow Charts**
- Work Flow Diagrams
- Data Flow Diagrams
- Wireframe / UI Mockups
- Infographics

Tools

- git and github
- Dropbox
- LaTeX/bibtex for the report
- Slack for chat rooms (Gophers/general)