Requirements and Expectations for Bachelor and Master Students

University of Passau
Faculty of Computer Science and Mathematics

Chair of Computer Engineering
Prof. Dr. Stefan Katzenbeisser

1 Introduction

This document helps students and supervisors to organize their communication in an optimal way. It illustrates the expectations for students and supervisors and gives hints for an effective cooperation and flow of information.

2 Preliminary Remarks

A thesis is a full-time job. Students are expected to work full-time on their thesis project. While this is demanding, the benefits are huge:

- The chance to become part of a research group — this can lead to involvement with other projects, social network, good friends, valuable resources, ...
- The chance to work with researchers on a real project.
- The end result of the student's project can be used for resume material that is available for all future employers to see.

3 Self-Motivation and Steady Schedule

The student is expected to be self-motivated. The supervisor may push the student to excel, but if the student is not self-motivated to work, then the student probably will not get much out of doing her thesis in our group. The student should schedule time to work on the project each day and keep to a regular schedule.

4 Organizational Issues

- The thesis has to be registered at the registrar's office at the earliest.
- At least one additional appointment with the supervisor is mandatory. The student has to tell the supervisor the agenda of the meeting early enough, so the supervisor can prepare for the meeting.

5 Frequent Communication with Supervisor

Frequent communication with your supervisor is a must. Your supervisor should have a good idea of:

- what you are currently working on,
- how far you have gotten,
- how you are implementing it,
- what you plan on working on next,
- what issues have come up,
- what you did to get around them,
- what is blocking you if you are stuck.

The supervisor likely has worked on related topics for long enough to know the history of decisions, how things are architected, the state-of-the-art, the other people involved, the process for doing things, and all other cultural lore that will help the student be most successful. Before the thesis project has started, the supervisor and student should iron out answers to the following questions:

1. What is the communication schedule? Mondays, Wednesdays, Fridays?
2. What is the best medium to use for regular, scheduled communication? E-mail? Telephone call? Face-to-face? Wiki?
3. What is the best medium to use for non-scheduled communication? E-mail? Wiki?

DO:

- Keep your supervisor up to date as much as possible.
- This forces you to be more organized and it gives your supervisor a chance to help you out if you are having trouble.
- Let your supervisor know what your schedule is.
- Are you going on vacation, moving, writing papers for class? If your supervisor does not know where you will be or when to expect a lag in your productivity, he/she could not help you correct or plan accordingly.

AVOID:

- Going for more than two weeks without communicating with your supervisor.

6 Programming

6.1 Technical Foundation

- The use of external libraries should be coordinated with the supervisor, to take care of license issues

6.2 Documentation

- All public interfaces must be documented comprehensively and private interfaces should also be documented. Classes should be commented, stating their purpose and the author.
- The relevant features of your code should be demonstrated with example code (if applicable), e.g. in the form of unit tests.

6.3 Version control

Students should use version control for their project.

DO:

- commit-early/commit-often
- This allows issues to be caught quickly and prevents the dreaded one-massive-commit-before deadline.
- Use quality commit messages.

- Bad examples:
  - Fixed a bug.
  - Tweaks.
- Good examples:
  - Fixed a memory leak where the thingamajig was not getting freed after the parent doohicky was freed.
  - Implemented Joe's good idea about rendering in a separate buffer and then swapping the buffer in after rendering is complete.
  - Improved HTML by simplifying tables.

AVOID:

- Checking in multiple non-related changes in one big commit.
- If something is bad about one of the changes and someone needs to roll it back, it is more difficult to do so.
- Checking in changes that have not been tested.

7 Documents

It is a good idea for the student to maintain documents during the course of the project. These documents should cover:

1. the thesis plan
2. deviations from the project plan and how and why the original plan changed
3. state-of-the-art
4. any issues that could not be worked out or overcome
5. possible future directions
6. any resources used or relevant specifications

8 Delivering your Thesis

You need to deliver:

- 2 bound versions for Bachelor Thesis, 3 bound versions for Master Thesis
- A digital (CD/DVD/USB/GIT) delivery containing:
  - Readme with a table of contents in the root folder
  - Executable software with a handbook, describing the usage
  - Sources of the software, with instructions regarding dependencies and compilation

- PDF and sources (DOC/ODF/Tex, pictures) of your thesis
- All references as PDF
- All data including description (database schemas, file formats)

9 Recommendations

9.1 General

- The thesis should be written in LaTeX
- A Bachelor thesis can be written in German or English.
- A Master thesis should be written in English.
- Documenting your steps helps everyone: supervisors have a better overview, and the student has accompanying notes for her/his thesis.

9.2 Writing

9.2.1 Structuring

The Master thesis has to be sensibly structured, which is reflected in the content and the sequence of the chapters. The individual chapters should be based on each other, which means thatthere should be no forward references, whereas references to preceding chapters relate the current text passage to previous results and thus emphasize the logically consistent structure of the thesis.

Title page: The layout of the title page has to be compliant to the specifications of the University of Passau. Examples can be found in our provided templates.

Table of contents: The table of contents lists the numbered titles of chapters and subchapters with their respective page number. The subdivision should not exceed four levels. One level should include at least two subchapters otherwise the level can be omitted.

Introduction: The thesis begins with an introduction leading to the subject and highlights the relevance of the subject, which should be explained clearly. In addition, you should illustrate the subject by using a clear example. Get back to this example in the chapter discussing the results to close the circle. An overview of the structure closes the thesis.

Introducing chapters: The thesis must be comprehensible without reading additional technical literature. This is why an explanation of relevant terms and aspects is necessary. The reader, who only has a general knowledge of the topic, is thus provided with important extra information. Another, very essential task of these chapters is to put the Master thesis in the context of the state of the art. The following questions are answered: Are there other works about this problem with other solution approaches? How was the problem solved until now? Why are my ideas new?

Main chapters: The core chapters should show the chosen working method and the achieved results in a well-structured way (top-down perspective). The formulation of the reasons for choice of this particular working method, the presentation of all significant details and the interpretation of the results are the most important aspects.

Details about the implementation of software are only relevant if they are crucial to the thesis.

Conclusion / Future Work: The thesis ends with a summary and an outlook. Essential findings are mentioned again without addressing the achievement of the results. The outlook contains information on what more could / would have been done if there had been more time. Thus the author can prove that they are able to realize generalizations of their solution or rather an application of their approach to other fields.

References: The thesis is embedded in relevant technical and research literature and refers to it by references in the text. The literature is listed in the references at the end of the thesis in alphabetical order by author or in numerical order.

Appendix: It is possible to add a list of abbreviations, a glossary or other appendices, such as technical documents for consulting, complete list or descriptions of characteristics (e.g. the syntax of a language, language definitions), extensive examples, essential algorithms, directions for use, intallation guide, large tables, such as performance data. These pieces of data do not fit into the chapter structure and disrupt the text. If some of these documents are not necessary for the understanding of the thesis or to round off the overall impression, they can be submitted over the digital submission (CD/DVD/USB/Git), along with the software documentation. The implemented program is not part of the appendix! Put the comprehensive documentation of the implementation in a separate document.

The introduction and the concluding chapter "Conclusion/Future Work" should also make sense when read without the chapters in between. They should make the reader curious to read the whole work.

9.2.2 Formulation and Style

- Start each chapter with a short overview of its content and objectives. Think carefully about them before you start writing.
- Sensibly subdivide the sections (except introduction and conclusion) in chapters and these, if necessary, in subchapters. The hierarchy should nor have too many levels. Chapters or subchapters should not be shorter than half a page.
- Avoid "phenomenological" listings such as enumerating descriptions of all constructs of a programming language or all functions of a system. These kinds of text belong in the appendix. The chapters are used to describe and illustrate concepts.
- Exemplify your points with suitable examples. The understanding of connections can often be improved by using a particular example consistently.
- Do not turn your thesis into a development history of the system (bottom-up style). Describe your result (top-down style) and point out alternatives.
- Avoid overlong and over complex sentences.
- Avoid complex participle constructions, such as "The efficiency losses determined after a thorough analysis of the algorithm were considerable."
- Avoid too many passive formulations
- Put a suitable, short caption under each figure/table. Interpret and reference the respective figure/table in the following paragraphs.

10 Presentation

10.1 My thesis in a nutshell

Describe on one slide:

- the motivation of your thesis (why is the problem worth of attention?)
- the main challenges you have handled (or you expect which occure)
- the main result

Why? In one slide you can not go into details, but the audience will know the context.

You should use an example, so that your audience has a picture in mind what you are talking about.

10.2 Agenda/Structure

Sho the structure of your talk. Most often this will be

- Motivation for my thesis
- Related Work
- Challenges/Research Questions/Technical Questions
- Methodology to solve the challenges/answer the questions
- Experiments/Implementation Details
- Discussion
- Conclusion/Future Work

Don't spend too long explaining the outline

10.3 Motivation

- Why is your work important?
- Why should I bother listening to you for 15 minutes of my valuable time?
- No motivation, no questions?

10.4 Related Work

- What did others do in the field? But only talk about **related** work, not about all work that has been done in the field.
- Summarise the work of others, do not repeat it
- What challenges have been already solved, which are still open?

10.5 Challenges/Research Questions

- What detailed questions/challenges did you plan to solve?
- What was the exact challenge/research question you solved?

- Maximum 2-3 slides
- Maximum 3-4 questions

10.6 Methodology

- How did you answer your questions
  - Started implementing immediately
  - Did a data model, then implemented it
  - Did experiment 1 to answer question 1, than followed by experiment 2 etc.
  - Be precise, be concise!
- If it is a starting talk, introduce how you plan to answer your questions

10.7 Implementation/Experiments

- Describe how you implemented the methodology or how you plan to do it (Software model, data model, implementation)
- Show experimental results or outline what you expect out of your experiments or how you proof your claims
  - Simulations
  - Data Analysis
  - Examples
  - Demos

10.8 Discussion

Discussion is a critical, objective reflection of what you achieved.

- Could you solve the challenge/answer the question? If yes, why. If no, why.
- What worked well? Where did you have problems.
- Be careful what you claim, most work does not solve a problem entirely, statements like "with my work I was able to overcome the Semantic Gap" will make knowledgeable audience very suspicious.

10.9 Conclusion/Future Work

- Summarize what you did for those that fell asleep although you did a great talk
- Point to open issues
  - No open issues, points towards a trivial work.