

Scientific Writing for a Final Year Thesis

Des Traynor

8th February 2006

Outline

- Preparing to Write Up
- How a thesis should be structured
- How to write in a scientific style

Disclaimer

- Before I start, I must point out that this is in no way affiliated with your final year project.
- It's basically my opinions of stuff that works well and gets good marks.
- I offer no guarantees that it will work for you.

Preparing for the Write Up

- Allow sufficient time, your thesis is worth a lot of your FYP marks
- At least 2 people will read all of your thesis, the same can't be said for your code.
- Get an outline for the thesis as early as possible. With a good outline, you only need to fill in the blanks.

Science or Software Engineering

- Every fourth year thesis usually involves scientific research or software development. Often they will involve both.
- The structure of a thesis is slightly different depending on whether code is there to answer a research question, or if the code is the project itself.

S.E Chapter 1 Introduction

- What problem does your system solve?
- Why is that an important problem?
- What other solutions have been used before?
- What makes your one different?
- Finally, provide an overview of the remaining chapters.

SE Chapter 2: Requirements

- What did you have to build?
 - List of requirements (numbered)
 - List of attributes
- Requirements Analysis
 - Was it all possible?
 - Were there any conflicting requirements?
 - How did this affect the project?

SE Chapter 3: Design

- How is your system designed?
- How many separate modules are there
 - Here is a good place for diagrams, flowcharts etc
- Did you use any well known design methodologies (e.g. model-view-controller)
- What are the benefits of your design?

SE Chapter 4:Implementation

- What tools could you use for the job?
 - e.g. JAVA, Matlab, php, whatever
- What tools did you use? Why?
 - Java, because its cross platform.
 - Matlab cause it can handle images well.
 - C++ to integrate with other software.
- Implementation issues can be discussed
 - i.e. If the language caused you hassle, here is a good place to explain why

SE Chapter 5: Testing

- How did you test your system
 - Unit testing, Integration Testing, White box testing, black box testing etc.
 - What actual tests did you perform?
 - What happened?
 - It's not bad to say you found serious faults with the system, its certainly better to declare them than to hide them.
- You will have to demo your software at some point, so lying isn't really a great idea.

SE Chapter 6: Conclusions

- This chapter should link with the first 2 chapters. You introduced a problem and you solved it.
- Basically this chapter should answer the question "So what?"
- **Always** have a future work section. It doesn't matter if you have no intention of doing it.

C.S Chapter1: Introduction

- What is your hypothesis?
- Why is your question interesting?
- What are you trying to achieve?

CS Chapter 2: Literature

- What have other people done?
- What new knowledge will your work add?
- What is the current state of the art, and where does your work fit?
- What body of work are you contributing to?

Chapter 3: Methodology

- What did you do to validating/analyse your hypothesis?
- What programs did you write, why?
- Is your method sound? Are there any factors you haven't included?
- Can your results be trusted?

Chapter 4: Results

- What did you find out?
- Is that what was expected?
- Here is where you give tables/stats etc
- Provide a detailed description of every significant result you have.
- You should also note anything that proved inconclusive, this is just as important.

Chapter 5: Conclusions

- So did you answer your question from the introduction?
- If so, how thoroughly?
- If not, why not?
- What further work can be done in this area?

Abstract

- Generally the abstract should be written at the very end.
- About 200 words
- It should give a good high level description of the report, enough to let someone know if they're interested.

So thats the outlines

- You can mix and match if you see fit, and change the order.
- No matter what you must have each section present in some form though.
- Try to get your outline finalized as soon as possible, it'll make everything easier.
- Onto Writing.

Scientific Writing

- Slightly different to normal English.
- Minimize opinions, maximize fact.
- Almost every sentence in the thesis should be...
 - True ("Java **is** a cross platform language")
 - Logical Conclusion ("Matlab has superior image processing capabilities than Java, therefore ...")
 - Cited (" Wireless networks have had many problems with security [Schneier, 05]")

Write to the point

- There should be very little ambiguity in your writing.
- Words like should/might/may be must be avoided. For the most part this are or aren't science.
- Avoid colloquial or relative language
 - JAVA is an easier language for GUIs (easier than what?)
 - It was correct to certain degree. (What degree?)

Other thoughts

- If you're working on an area that your supervisor has published in, you should read at least one of the papers (and cite them all)
- Try to get a good few references (10-30) into your thesis, it makes the work appear grounded.
- Ask your supervisor are they willing to read a draft, if so, you **must** get one to them. It's like free marks

Thats all

- Best of luck with your thesis.
- If you want these slides or the handouts, they are available here:
 - <http://www.minds.may.ie/~dez/sw-talk/>
- Thanks for your attention.