TDDD89

Lecture 2

First, some notes on submitting your thesis plans and the upcoming seminar.

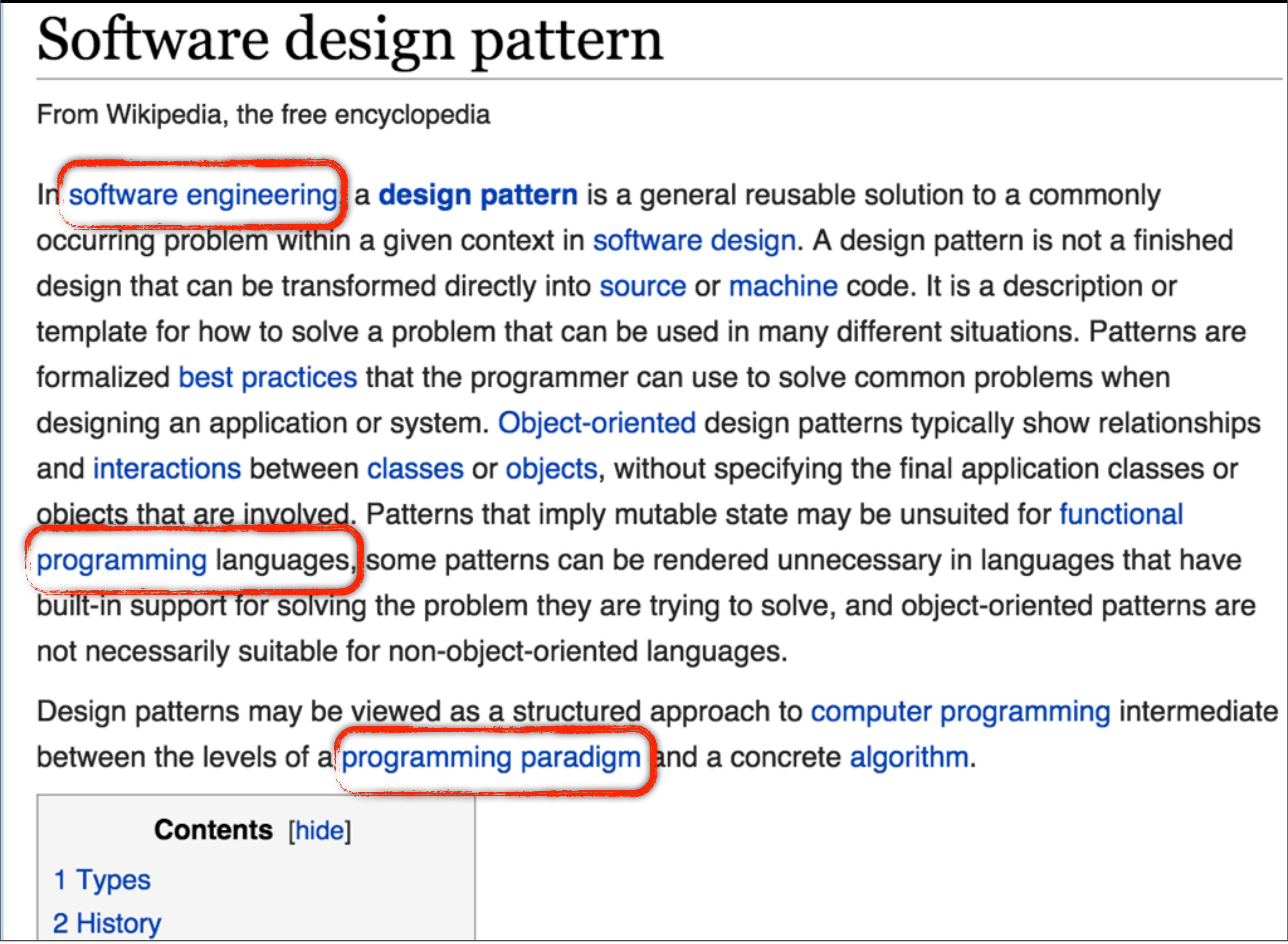
Course instructions

Information search

Finding information

fredag 6 november 15

Learn about the subject area:
use Wikipedia, books and previous course material
Extract keywords that you can use when searching papers.
Use Google Scholar & Unisearch first, specific publications second



Software design pattern

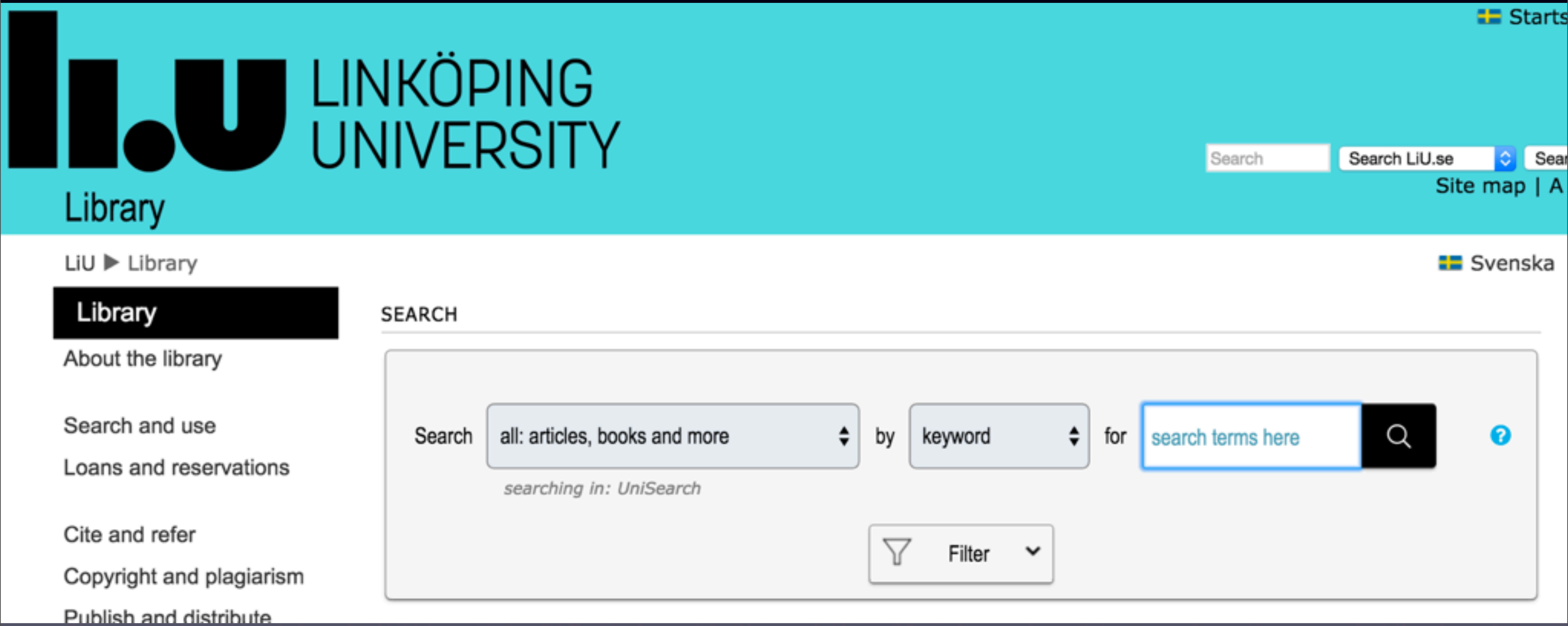
From Wikipedia, the free encyclopedia

In software engineering, a **design pattern** is a general reusable solution to a commonly occurring problem within a given context in software design. A design pattern is not a finished design that can be transformed directly into source or machine code. It is a description or template for how to solve a problem that can be used in many different situations. Patterns are formalized best practices that the programmer can use to solve common problems when designing an application or system. Object-oriented design patterns typically show relationships and interactions between classes or objects, without specifying the final application classes or objects that are involved. Patterns that imply mutable state may be unsuited for functional programming languages, some patterns can be rendered unnecessary in languages that have built-in support for solving the problem they are trying to solve, and object-oriented patterns are not necessarily suitable for non-object-oriented languages.

Design patterns may be viewed as a structured approach to computer programming intermediate between the levels of a programming paradigm and a concrete algorithm.

Contents [hide]

Stå på giganters axlar



Starts

LINKÖPING UNIVERSITY

Library

Search | Search LiU.se | Sea

Site map | A

LiU ▶ Library

Svenska

Library

About the library

Search and use

Loans and reservations

Cite and refer

Copyright and plagiarism

Publish and distribute

SEARCH

Search [all: articles, books and more] by [keyword] for [search terms here]

searching in: UniSearch

Filter

How to find information

”HLA active probing runtime performance requirements in a Wide Area Network”

→

{ }

HLA

simulation

active probing

fault detection and localization

runtime performance requirements

latency, throughput

in a Wide Area Network

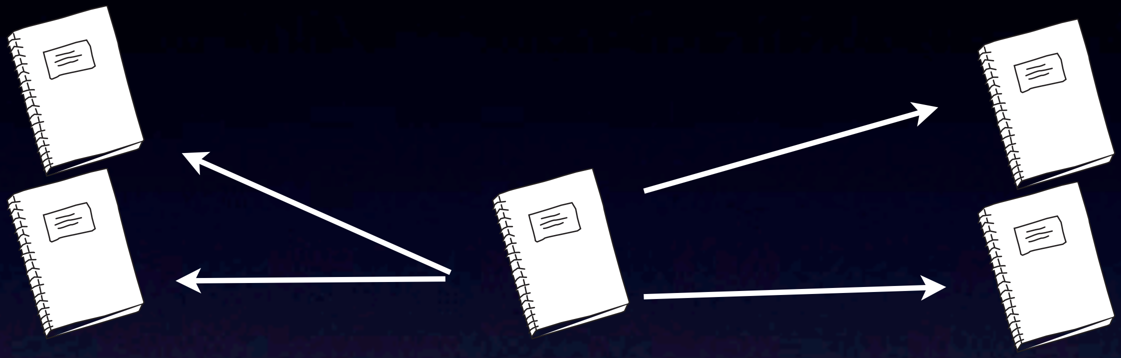
IP networks

Engineering information vs Scientific information

	Engineering	Science
Questions	How to solve a problem	How to explain something
Reliability	Working solutions, proven theories	Cited work
Sources	White papers, software projects	Reviewed publications

Iterative search

”On Heuristic Search”



”On $\boxed{\text{AI}}$”
”On Planning”

”$\boxed{\text{Fie}}$ Planner”

”$\boxed{\text{Foo Heuristic}}$”

1

2

3

Scientific publishing

Primary studies

Secondary studies

Text books

”What”

Scientific publishing

Peer-reviewed publications

- Conference Proceedings
- Journal papers

Non-reviewed publications

- Technical reports
- White papers

"How"

Publication types

P. Kruchten, H. Obbink, and J. Stafford. The past, present, and future for software architecture. *IEEE Software*, 23(2):22–30, March–April 2006.

No empirical results. Shares experience on Software Architecture research and development.

Publication types

T. K. Paul and M. F. Lau. A systematic literature review on modified condition and decision coverage. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, SAC '14, pages 1301–1308, New York, NY, USA, 2014. ACM.

Systematic Literature Review, secondary study. Published at a conference

Publication types

C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering*. Springer Berlin Heidelberg, 2012.

Guidelines textbook on empirical methods in Software Engineering

Publication types

I. Maier, T. Rompf, and M. Odersky. Deprecating the observer pattern. Technical report, École Polytechnique Fédérale de Lausanne, 2010.

Technical report, non-reviewed publication. No empirical support for claims, but suggestions of an architecture.

Publication types

A. Nilsson, J. Bosch, and C. Berger. Visualizing testing activities to support continuous integration: A multiple case study. In G. Cantone and M. Marchesi, editors, *Agile Processes in Software Engineering and Extreme Programming*, volume 179 of *Lecture Notes in Business Information Processing*, pages 171–186. Springer International Publishing, 2014.

Case study, reviewed publication in journal

Publication types

J. Andrews, L. Briand, and Y. Labiche. Is mutation an appropriate tool for testing experiments? In *Proceedings of the 27th International Conference on Software Engineering*, ICSE 2005, pages 402–411, May 2005. IEEE Computer Society.

Experiment, reviewed publication presented at a conference and published in proceedings from the conference.

What are results?

Type	How?	Quality
Procedure/ technique	Formal proofs, experiments, statistical support,	Proper use of statistics
Descriptive Models		Properly accounting for reality
Experience reports	Interviews, observations, usage data	Real systems & people

What are *strong* results?

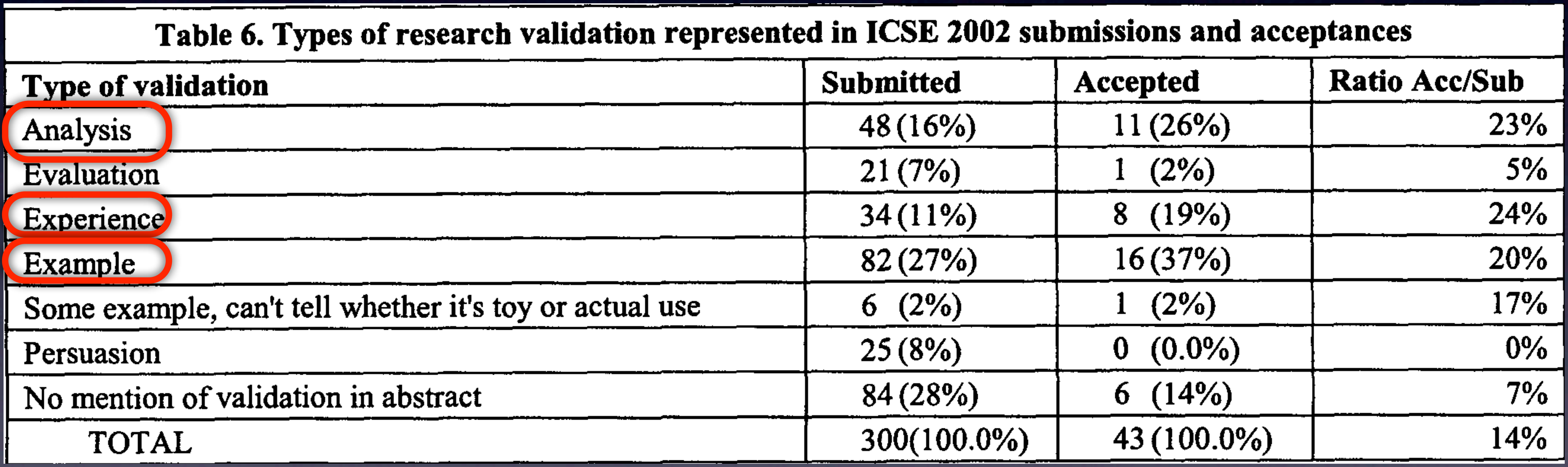


Table 6. Types of research validation represented in ICSE 2002 submissions and acceptances

Type of validation	Submitted	Accepted	Ratio Acc/Sub
Analysis	48 (16%)	11 (26%)	23%
Evaluation	21 (7%)	1 (2%)	5%
Experience	34 (11%)	8 (19%)	24%
Example	82 (27%)	16 (37%)	20%
Some example, can't tell whether it's toy or actual use	6 (2%)	1 (2%)	17%
Persuasion	25 (8%)	0 (0.0%)	0%
No mention of validation in abstract	84 (28%)	6 (14%)	7%
TOTAL	300(100.0%)	43 (100.0%)	14%

M. Shaw. Writing good software engineering research papers: Minitutorial. In *Proceedings of the 25th International Conference on Software Engineering*, ICSE '03, pages 726–736, Washington, DC, USA, 2003. IEEE Computer Society.

Strong results

Real systems &&
proper analysis

How to evaluate papers

- Relevance = f(title, year, abstract, citations)
- The more specific the paper, the less citations?
- Literature reviews: meta studies
- Publication types: journals, conferences, book chapters
- Refer to the main results of the paper, not that which is written in the introduction

What about white papers/other stuff?

- Use to support existence: ”There are several implementations of Flux controllers”
- Not to support claims and propositions: ”Flux controllers are more user friendly than Flax controllers”

Evaluation of paper

”Software product lines are related software products that are customized to different customers [1]”

[1] Kästner, C., Apel, S., and Kuhlemann, M. Granularity in software product lines. In *Proceedings of the 30th International Conference on Software Engineering*, ICSE ’08, pages 311–320, New York, USA, 2008.

Not the main result of [1]

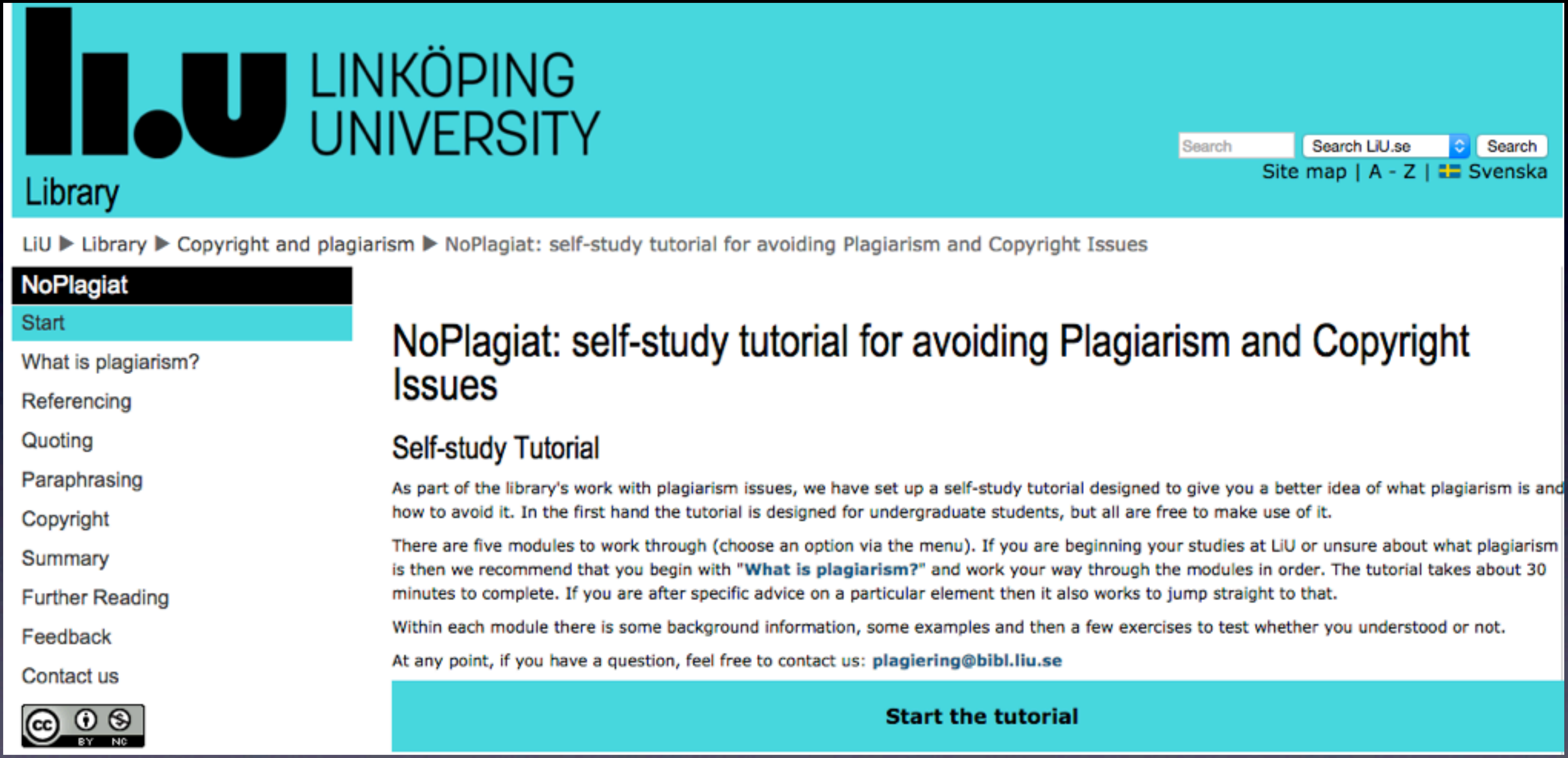
[1] Pohl, K., Böckle, G., and van der Linden, F. J. (2005). *Software product line engineering: foundations, principles and techniques*. Springer Science & Business Media.

Use above reference instead

Theory

1. **Analysis**: what is this? Classifications, taxonomies, ontologies
2. **Explanations**: why does something happen?
3. **Predictions**: what will happen?

Plagiarism & copyright



NoPlagiat: self-study tutorial for avoiding Plagiarism and Copyright Issues

Self-study Tutorial

As part of the library's work with plagiarism issues, we have set up a self-study tutorial designed to give you a better idea of what plagiarism is and how to avoid it. In the first hand the tutorial is designed for undergraduate students, but all are free to make use of it.

There are five modules to work through (choose an option via the menu). If you are beginning your studies at LiU or unsure about what plagiarism is then we recommend that you begin with "**What is plagiarism?**" and work your way through the modules in order. The tutorial takes about 30 minutes to complete. If you are after specific advice on a particular element then it also works to jump straight to that.

Within each module there is some background information, some examples and then a few exercises to test whether you understood or not.

At any point, if you have a question, feel free to contact us: **plagiering@bibl.liu.se**

noplagiat.bibl.liu.se

Using image without reference

Plagiarism + copyright violation

Using image with reference

Copyright violation

Using own/CC image with reference

OK!

Using references

References

~~[1] has studied software design patterns~~

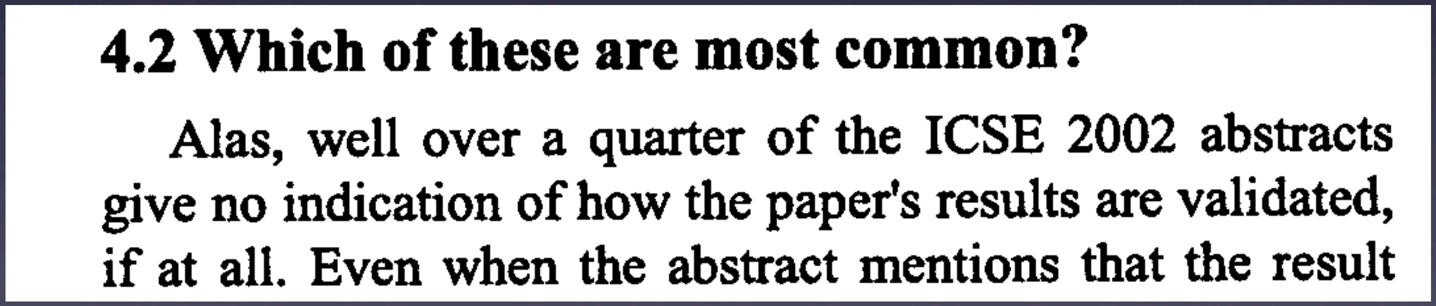
Odersky et al. have studied software design patterns [1].

Odersky et al. (2010) have studied software design patterns.

There are a number of conventions of how to use references properly: use in-text references or outside-text references consistently. IEEE has a good standard for this.

Paraphrasing

~~Over a quarter of the ICSE 2002 abstracts give no indication of how the paper's results are validated, if at all [1].~~



4.2 Which of these are most common?

Alas, well over a quarter of the ICSE 2002 abstracts give no indication of how the paper's results are validated, if at all. Even when the abstract mentions that the result

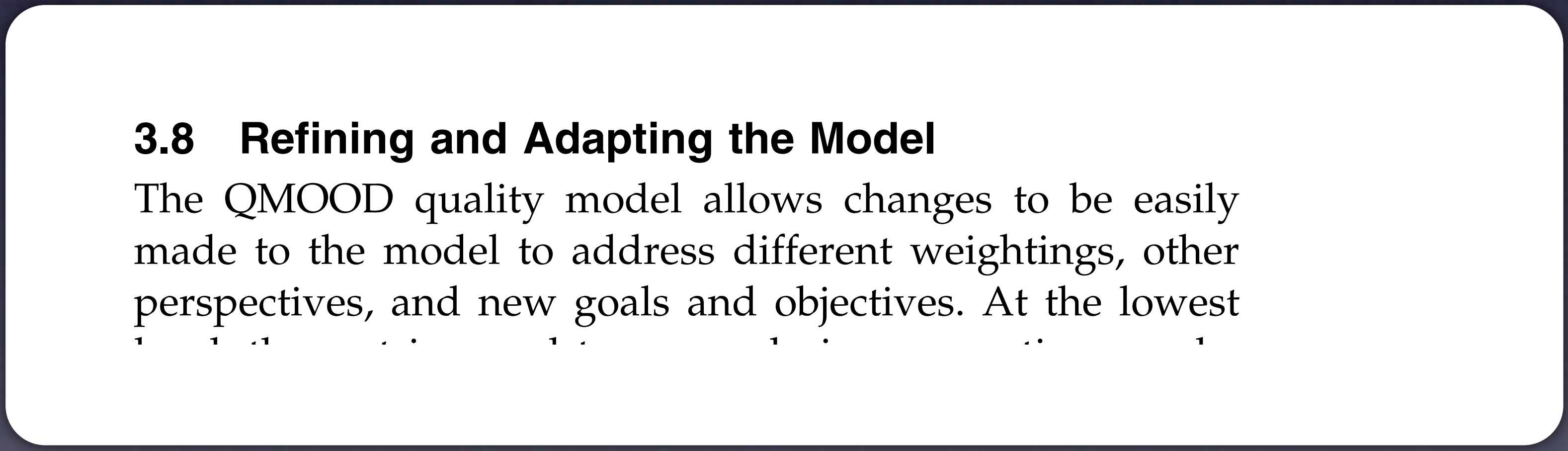
[1] M. Shaw. Writing good software engineering research papers: Minitutorial. In *Proceedings of the 25th International Conference on Software Engineering*, ICSE '03, pages 726–736, Washington, DC, USA, 2003. IEEE Computer Society.

fredag 6 november 15

Do not copy verbatim from published papers.

Citations

Bansiya and Davis claim that the QMOOD model may address ”different weightings, other perspectives, and new goals and objectives” [1]



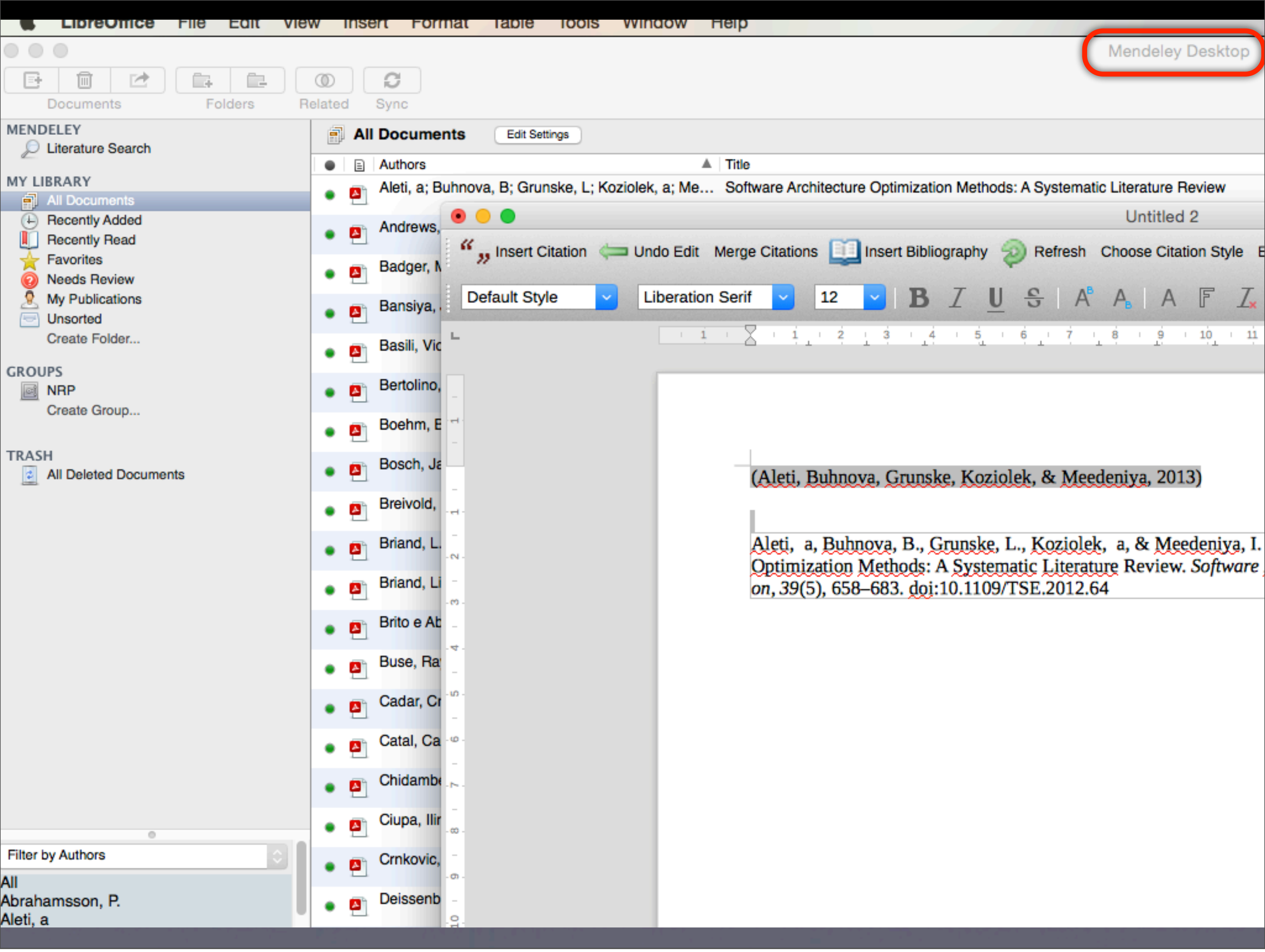
3.8 Refining and Adapting the Model

The QMOOD quality model allows changes to be easily made to the model to address different weightings, other perspectives, and new goals and objectives. At the lowest

[1] J. Bansiya and C. Davis. A hierarchical model for object-oriented design quality assessment. *IEEE Transactions on Software Engineering*, 28(1):4–17, Jan 2002.

Use proper citation if needed, but only cite if necessary.

Managing references

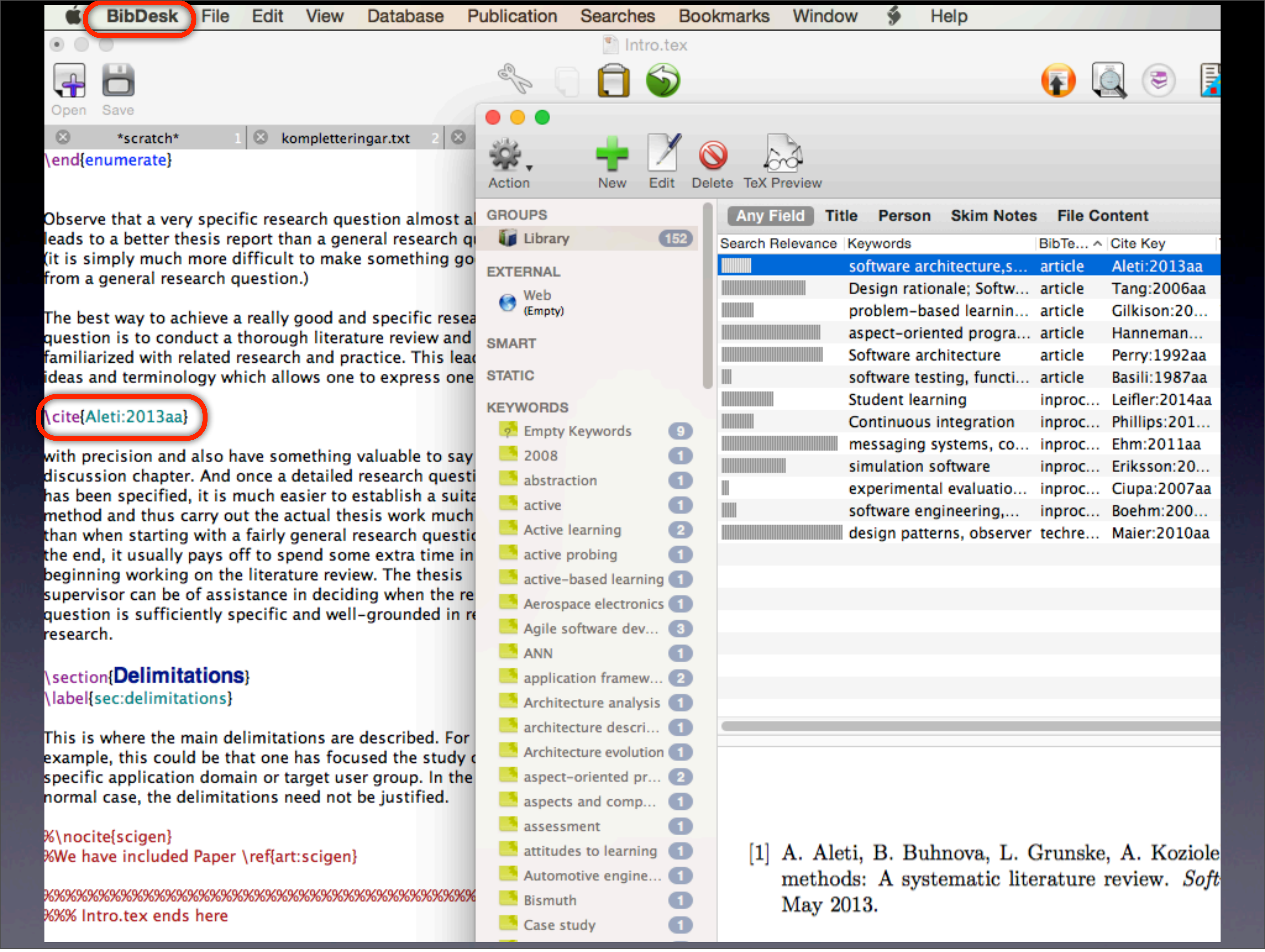


Authors	Title
Aleti, a; Buhnova, B; Grunske, L; Koziolek, a; Me…	Software Architecture Optimization Methods: A Systematic Literature Review
Andrews,	
Badger, M	
Bansiya,	
Basili, Vic	
Bertolino,	
Boehm, E	
Bosch, Ja	
Breivold,	
Briand, L.	
Briand, Li	
Brito e Ab	
Buse, Ra	
Cadar, Cr	
Catal, Ca	
Chidambe	
Ciupa, Ilir	
Crnkovic,	
Deissenb	

(Aleti, Buhnova, Grunske, Koziolek, & Meedeniya, 2013)

Aleti, a, Buhnova, B., Grunske, L., Koziolek, a, & Meedeniya, I.
Optimization Methods: A Systematic Literature Review. *Software*
on, *39*(5), 658–683. doi:10.1109/TSE.2012.64

fredag 6 november 15



```
\end{enumerate}

Observe that a very specific research question almost al
leads to a better thesis report than a general research q
(it is simply much more difficult to make something go
from a general research question.)

The best way to achieve a really good and specific resea
question is to conduct a thorough literature review and
familiarized with related research and practice. This lea
ideas and terminology which allows one to express one

\cite{Aleti:2013aa}

with precision and also have something valuable to say
discussion chapter. And once a detailed research questi
has been specified, it is much easier to establish a suita
method and thus carry out the actual thesis work much
than when starting with a fairly general research questi
the end, it usually pays off to spend some extra time in
beginning working on the literature review. The thesis
supervisor can be of assistance in deciding when the re
question is sufficiently specific and well-grounded in re
research.

\section{Delimitations}
\label{sec:delimitations}

This is where the main delimitations are described. For
example, this could be that one has focused the study o
specific application domain or target user group. In the
normal case, the delimitations need not be justified.

%\nocite{scigen}
%We have included Paper \ref{art:scigen}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Intro.tex ends here
```

Keywords	Count
Empty Keywords	9
2008	1
abstraction	1
active	1
Active learning	2
active probing	1
active-based learning	1
Aerospace electronics	1
Agile software dev...	3
ANN	1
application framew...	2
Architecture analysis	1
architecture descri...	1
Architecture evolution	1
aspect-oriented pr...	2
aspects and comp...	1
assessment	1
attitudes to learning	1
Automotive engine...	1
Bismuth	1
Case study	1

Search Relevance	Keywords	BibTe...	Cite Key
▮▮	software architecture,s...	article	Aleti:2013aa
▮▮▮▮▮	Design rationale; Softw...	article	Tang:2006aa
▮▮	problem-based learnin...	article	Gilkison:20...
▮▮▮▮▮▮	aspect-oriented progra...	article	Hanneman...
▮▮▮▮▮▮	Software architecture	article	Perry:1992aa
▮	software testing, functi...	article	Basili:1987aa
▮▮▮	Student learning	inproc...	Leifler:2014aa
▮▮	Continuous integration	inproc...	Phillips:201...
▮▮▮▮▮▮▮	messaging systems, co...	inproc...	Ehm:2011aa
▮▮▮▮	simulation software	inproc...	Eriksson:20...
▮	experimental evaluatio...	inproc...	Ciupa:2007aa
▮	software engineering,...	inproc...	Boehm:200...
▮▮▮▮▮▮▮	design patterns, observer	techre...	Maier:2010aa

[1] A. Aleti, B. Buhnova, L. Grunske, A. Koziole
methods: A systematic literature review. *Soft*
May 2013.

Writing about what you've read

- Take notes of what you've read
- Consider what needs to be in your report. Do not write everything you've read in your report. Remember to have a strong connection to your main method/results

Summary

- Start learning about the subject, then find proper support for your claims. Use different sources for learning and references.
- There are different types of academic publications and results.
- Do not plagiarize or copy images.
- Use proper reference management software (check course web)