How to Write a Great Guided Research
And why should I do it?

Roman Haas

With material from Dr. Elmar Juergens

In close cooperation with the Academic Advisors at TUM Informatics
Questions?

• Please join our discussion sessions
  – Bachelor’s / Master’s Thesis 17.06. 11:00 am
  – Guided Research 17.06. 2:30 pm

• For details, please refer to moodle
thesisguide.org

- Slides
- Video
- Detailed Essays
- FAQ
Agenda

1. Motivation
2. Preparation
3. Doing the work
Guided Research

• Guidance
  – Advisor has research experience, helps you on your way
  – Examiner must be from TUM Informatics or affiliated with the Department of Informatics

• Your own (small) research project
  – Related Work
  – Implementation?
  – Proof?
  – Evaluation?

• Document and present your work

➢ Insights into real scientific work
<table>
<thead>
<tr>
<th>Guided Research</th>
<th>Master’s Thesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Voluntary</td>
<td>• Mandatory</td>
</tr>
<tr>
<td>• 6 months, 10 ECTS</td>
<td>• 6 months, 30 ECTS</td>
</tr>
<tr>
<td>• Effort comparable to a more labor-intensive lab course</td>
<td>• Full-Time</td>
</tr>
<tr>
<td>• Approx. 40 students/semester</td>
<td>• Approx. 100 students/semester</td>
</tr>
</tbody>
</table>
Less Formal than a Thesis

• Written document is „just“ a scientific report on your results (8-12 pages in English) which you need to send to your advisor/examinor only

• You have to present your work
  – At the chair
  – Or at a „scientific event“
There are some formalia, though...

- You have to be enrolled in a Master’s program (Informatics, Data Engineering & Analytics, Information Systems, Games Engineering)

- Registration must be done in the first lecture week
- Submission no later than the first lecture week of the next semester (6 months duration)
- Cannot be extended

- No transfer of credits, you need an internal examiner (with whom you may work together abroad)
Learning to Rank Extract Method Refactoring Suggestions for Long Methods

• Given: A set of refactoring suggestions for long methods

• Needed: scoring function to find out which is the best

• Approach: machine learning
1. Introduction

A long-term goal in the field of software engineering is to develop methods that can automatically extract software metrics from code. This is an important task because software metrics are widely used in various areas, such as software quality assurance, project management, and software process improvement.

1.1. Motivation

Software metrics are essential for understanding the quality, size, complexity, and other characteristics of software systems. They are used in various aspects of software engineering, such as defect prediction, cost estimation, and project planning. However, the manual extraction of software metrics is a time-consuming and labor-intensive task. Therefore, the development of automatic methods for extracting software metrics is highly desired.

1.2. Previous Work

Previous research has focused on various aspects of software metrics extraction. Some studies have proposed methods for extracting software metrics from source code using natural language processing techniques. Others have developed machine learning models to predict software metrics based on code characteristics. However, these methods often suffer from low accuracy and poor generalization.

1.3. Contributions

In this paper, we present a new method for automatic software metrics extraction. Our method uses neural machine translation models to translate source code into a formal language, which can then be used to extract software metrics. We evaluate our method on a large dataset of software projects and show that it outperforms existing methods.

2. Methodology

2.1. System Overview

Our method consists of two main components: code translation and metric extraction. The code translation component uses a neural machine translation model to translate source code into a formal language, while the metric extraction component uses a set of predefined rules to extract software metrics from the translated code.

2.2. Code Translation

In this section, we describe how we use neural machine translation models to translate source code into a formal language. We employ a sequence-to-sequence model with attention mechanisms, which has been shown to be effective for machine translation tasks.

2.3. Metric Extraction

Once the code has been translated into a formal language, we use a set of predefined rules to extract software metrics. These rules are designed to capture the main characteristics of the code, such as the number of functions, the size of the code, and the complexity of the code.

3. Results

We evaluate our method on a large dataset of software projects and show that it outperforms existing methods. Our results demonstrate the effectiveness of our approach for automatic software metrics extraction.

4. Conclusion

In conclusion, we have presented a new method for automatic software metrics extraction. Our method uses neural machine translation models to translate source code into a formal language, which can then be used to extract software metrics. We believe that this approach has the potential to revolutionize the field of software metrics extraction and provide significant benefits for software engineering practices.

Acknowledgments

This work was supported by the National Natural Science Foundation of China. We would like to thank the anonymous reviewers for their constructive feedback on our paper.
Konferenz Tag 1 - 18.01.2017

Inhibition area / Kaffeepause & Networking im Ausstellungsbereich

München Wien
Salzburg Österreich

Continuous Integration für Mobile Apps

Renotre in the Agile World

Testumgebungen auf einen Klick - zentrales Testumgebungsmangement als Herausforderung und

1. Design und Wien (AT)
2. Design und Wien (AT)

Continuous Delivery - Realize your Quality - Every Day

Gebäude, Bewertung

A portfolio of internal quality Scrum in Embedded Systems

(Handels- und Wirtschaftsuniversität Wien)

Englisch, Fortgeschrittene

Learning to Rank: Extract Method Refactoring: Suggestions for Large Methods

(Technical University of Munich, Munich, DE)

Englisch, Fortgeschrittene
Chronological Overview

- Registration: 01/15
- SS 2015: 04/15
- Deadline TUM: 10/15
- Deadline SWQD: 04/16
- Camera-Ready Version: 08/16
- Deadline Slides: 01/17
What is Different to Other Study Projects?

• More Freedom
  – Topic
  – Own research
  – You define schedule and pace

• Requires high level of self-organization

• Better opportunities for personal growth
Personal Conclusion

- My GR was on my „mental Stack“ during my entire studies in the Master’s program
- GR got me out of my comfort zone
- Learned a lot on research methodologies and practical application of machine learning techniques
- Working on my research topic was fun for me
- I would do it again 😊
Timo Pawelka
Automatische Erkennung der Sprache von Quelltext-Kommentaren
Bachelor's Thesis, not published

Timo Pawelka
Elmar Juergens:
Is This Code Written in English? A Study of the Natural Language of Comments and Identifiers in Practice.
Design and Evaluation of Regression Test Suite Minimization Techniques
Master’s Thesis

Raphael Nömmer, Roman Haas
Test Suite Minimization
Guided Research, to be published in Conference Proceedings of SWQD ‘20
Funding

Costs 1k€ – 5k€
- Travel and accommodation costs
- Conference fee

Funding sources (often mixed)
- Travel Subsidies
- Chairs
- DAAD scholarships
- CQSE

Decision processes take long, so organize this early!
Agenda

1. Motivation

2. Preparation

3. Doing the work
Get the Most out of your GR?! 

- GR provides the opportunity to publish scientific work at a scientific venue.

- Nevertheless, formally, you do not need to publish anything

- My recommendation: aim for a scientific publication
<table>
<thead>
<tr>
<th>Track A</th>
<th>Track B</th>
<th>Track C</th>
<th>Scientific Track</th>
<th>Solution Provider Forum I</th>
<th>Solution Provider Forum II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous Integration for Mobile Apps</td>
<td>Improve your software models with search-based techniques</td>
<td>Renate in the Agile World</td>
<td>Testumgebungen auf einen Blick - zeitsparendes Testumgebungsmanagement als Herausforderung und</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Quality Days</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

14:05

**Continuous Delivery - Feel your Quality - Every Day**

(automate Software GmbH, Wien, AT)

<table>
<thead>
<tr>
<th>14:25</th>
</tr>
</thead>
</table>

**Strukturierte Tests bei dezentraler Dokumentation**

Wie man zwei Plagen mit einer Klappe schlägt

(berge Security Research & Consulting GmbH, Dann, DE)

<table>
<thead>
<tr>
<th>14:05</th>
</tr>
</thead>
</table>

**England, Fortgeschrittene**

Learning to Rank Extract Method Refactoring
Suggestions for Large Methods

(Technical University of Hamburg-Harburg, DE)

| 14:25 |

English, Fortgeschrittene

A portfolio of internal quality metrics for software architects

(University of Gothenburg, Göteborg, SE)

| 14:05 |

**Zertifizierung Quality Engineer für das Internet der Dinge**

(software Quality Lab GmbH, Linz, AT)

| 14:25 |

English, Fortgeschrittene

Validating Java via Symbolic Execution

(ZT Preisegger, Wien, AT)

| 14:05 | 14:25 |

Deutsch, Fortgeschrittene

Kontinuierliche Architekturanalyse

(Software Quality Lab, Linz)

14:05

14:25
Submissions  Selection Procedure  Agenda

10-50%
## Pecking Order

**Conference**  
10% - 25%

**Workshop**  
40% - 60%

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Full Name</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHASE</td>
<td>11th International Workshop on Cooperative and Human Aspects of Software Engineering</td>
<td>27-May</td>
</tr>
<tr>
<td>CSI-SE</td>
<td>5th International Workshop on Crowd Sourcing in Software Engineering</td>
<td>27-May</td>
</tr>
<tr>
<td>MET</td>
<td>International Workshop on Metamorphic Testing</td>
<td>27-May</td>
</tr>
</tbody>
</table>

Aim: Submission to workshops
Call for Papers

12th International Workshop on Software Clones (IWSC 2018)
Co-located with the 25th IEEE International Conference on Software Analysis, Evolution, and Reengineering (SANER 2018)
March 20, 2018, Campobasso, Italy

Software clones are often a result of copying and pasting as an act of addressing by programmers, and can occur at many levels: from simple statement sequences to blocks, classes, modules, models, requirements or architecture today.

IWSC series of events has provided a venue for researchers to present their results in their respective research areas.
In particular, we expect the in-depth discussion about IWSC 2018 are here on this page.

**SUBMISSION:**

Papers must conform to the IEEE proceedings paper format guidelines. If the paper is accepted, at least one author must attend the workshop and present the paper. Accepted papers will be published in the IEEE Xplore Digital Library along with the SANER proceedings.

All submissions must be in PDF and must be submitted online by the deadline via the IWSC 2018 EasyChair conference management system.

Submit your papers here >>> EasyChair <<<

**TOPICS OF INTEREST:**

Topics of interest include but not limited to:
- Use cases for clones and clone mining
- Experiences with clones and clone mining
- Types and nature of clones
- Causes and effects of clones
- Techniques and algorithms for clone detection
- Clone and clone pattern mining
- Tools and systems for detecting clones
- Applications of clone detection
- System architecture and clone mining
- Effect of clones to software quality
- Clone analysis in families of software
- Measures of code similarity
- Economic and trade-off models
- Evaluation and benchmarking
- Licensing and plagiarism issues
- Clone-aware software design
- Refactoring through clone detection
- Higher-level clones in models
- Clone evolution and variation
- Role of clones in software development

**IMPORTANT DATES:**

- Abstract submission deadline: January 19, 2018 AoE
- Paper submission deadline: January 26, 2018 AoE
- Notifications: February 16, 2018
- Camera Ready deadline: ** February 22, 2018 **
- Workshop day: March 20 2018

**GENERAL CHAIR:**

TBD

**PROGRAM CO-CHAIRS:**

- Ying (Jenny) Zou (ying.zou@queensu.ca), Queen's University, Canada
- Matthew Stephan (stephamd@miamioh.edu), Miami University, USA

**STEERING COMMITTEE:**

- James R. Cordy, Queen's University, Canada
- Katsuro Inoue, Osaka University, Japan
- Rainer Koschke, University of Bremen, Germany

Each paper will be reviewed by at least three members of the program committee following a full double-blind process. Authors must adhere to SANER's double blind guidelines - [http://saner.unimol.it/reatrack](http://saner.unimol.it/reatrack). The following types of papers are sought:

- Full papers (7 pages maximum)
- Position papers (2 pages maximum)
- Tool demonstration papers (4 pages maximum)
Call for Papers

12th International Workshop on Software Clones (IWSC 2018)
Co-located with the 25th IEEE International Conference on Software Analysis, Evolution, and Reengineering (SANER 2018)
March 20, 2018, Campobasso, Italy

Software clones are often a result of copying and pasting as an act of addressing by programmers and can occur at many levels, from simple statement sequences to blocks, classes, programs, models, requirements or architectural artifacts.

IWSC series of events has provided a focal point for practitioners and researchers to share experiences, and in particular, we expect the in-depth discussions on clone detection and characterization that ISWC 2018 are here on the cutting edge.

TOPICS OF INTEREST:

Topics of interest include but not limited to:
- Use cases for clones and clone detection
- Experiences with clones and clone detection
- Types and nature of clones
- Causes and effects of clones
- Techniques and algorithms for clone detection
- Tools and systems for detecting clones
- Applications of clone detection
- System architecture and clone detection
- Effect of clones to system design
- Clone analysis in families and evolution
- Measures of code similarity
- Economic and trade-off models for clone decision
- Evaluation and benchmarking tools
- Licensing and plagiarism detection
- Clone-aware software design
defining and applying refactoring through clones
- Higher-level clones in modern systems
- Clone evolution and variation
- Role of clones in software engineering

PAPERS SOUGHT:

Each paper will be reviewed by at least three members of the program committee following a full double-blind process. Authors must adhere to SANER's double blind guidelines - http://saner.unimore.it/retrack. The following types of papers are sought:

- Full papers (7 pages maximum)
- Position papers (2 pages maximum)
- Tool demonstration papers (4 pages maximum)

SUBMISSION:

Papers must conform to the IEEE proceedings paper format guidelines. If the paper is accepted, at least one author must attend the workshop and present the paper. Accepted papers will be published in the IEEE Xplore Digital Library along with the SANER proceedings.

All submissions must be in PDF and must be submitted online by the deadline via the IWSC 2018 EasyChair conference management system.

Submit your papers here >>> EasyChair <<<
# Program Committee

<table>
<thead>
<tr>
<th>Name</th>
<th>Institution</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toshihiro Kamiya</td>
<td>Shimane University</td>
<td>Japan</td>
</tr>
<tr>
<td>Daping Hou</td>
<td>Clarkson University</td>
<td>USA</td>
</tr>
<tr>
<td>Tien Nguyen</td>
<td>University of Texas at Dallas</td>
<td>USA</td>
</tr>
<tr>
<td>Nils Göde</td>
<td>CQSE GmbH</td>
<td>Germany</td>
</tr>
<tr>
<td>Jens Krinke</td>
<td>University College London</td>
<td>UK</td>
</tr>
<tr>
<td>Otavio Lemos</td>
<td>ICT-UNIFESP</td>
<td>Brazil</td>
</tr>
<tr>
<td>Manishankar Mondal</td>
<td>University of Saskatchewan</td>
<td>Canada</td>
</tr>
<tr>
<td>Ravindra Naik</td>
<td>Tata Consultancy Services</td>
<td>India</td>
</tr>
<tr>
<td>Robert Tairas</td>
<td>Vanderbilt University</td>
<td>USA</td>
</tr>
<tr>
<td>Minhaz Zibran</td>
<td>University of New Orleans</td>
<td>USA</td>
</tr>
<tr>
<td>Eunjong Choi</td>
<td>Nara Institute of Science and Technology</td>
<td>Japan</td>
</tr>
<tr>
<td>Michael Godfrey</td>
<td>University of Waterloo</td>
<td>Canada</td>
</tr>
<tr>
<td>Yoshiki Higo</td>
<td>Osaka University</td>
<td>Japan</td>
</tr>
<tr>
<td>Foutse Khomh</td>
<td>Ecole Polytechnique de Montréal</td>
<td>Canada</td>
</tr>
<tr>
<td>Nicholas A. Kraft</td>
<td>ABB Corporate Research</td>
<td>USA</td>
</tr>
<tr>
<td>Chanchal Roy</td>
<td>University of Saskatchewan</td>
<td>Canada</td>
</tr>
<tr>
<td>Hitesh Sajnani</td>
<td>Microsoft</td>
<td>USA</td>
</tr>
<tr>
<td>Suresh Thummalapenta</td>
<td>Microsoft</td>
<td>USA</td>
</tr>
<tr>
<td>Xiyouin Wang</td>
<td>University of Texas at San Antonio</td>
<td>USA</td>
</tr>
<tr>
<td>Norhiro Yoshida</td>
<td>Nagoya University</td>
<td>Japan</td>
</tr>
</tbody>
</table>
What If I have no Topic in Mind?

- Ask potential advisors for ideas
  - Advisor from Bachelor’s Thesis
  - Lectures
  - Seminars
  - Lab courses

- As an advisor, I do **not** expect
  - Students to come up with thesis topics
  - Students to apply only for documented topics

- If you have a rough idea, discuss it with potential advisors
Requirements for a GR topic

- Is there a clear problem statement?
- Can different solutions be evaluated objectively?

Why?
- Decision making while you work on it
- Easier to convince advisor
- Easier to convince program chair

Even more important for a GR than BA/MA

More info: www.thesisguide.org
What Makes a Good Guided Research Advisor

- Needs to have publishing experience
- Has already successfully published (ideally on the same workshop if you aim for a publication)
- Sources: scholar.google.com, DBLP, personal webpage.

Elmar Juergens
CGSE GmbH
Bestätigte E-Mail-Adresse bei cgse.eu - Startseite
Software Qualität

<table>
<thead>
<tr>
<th>TITEL</th>
<th>ZITIERT VON</th>
<th>JAHR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do code clones matter?</td>
<td>375</td>
<td>2009</td>
</tr>
<tr>
<td>COPE-automating coupled evolution of metamodels and models</td>
<td>198</td>
<td>2009</td>
</tr>
<tr>
<td>Clone detection in automotive model-based development</td>
<td>172</td>
<td>2008</td>
</tr>
</tbody>
</table>
Agenda

1. Motivation

2. Preparation

3. Doing the work
View as an Advisor

- Regular meeting
- Meeting on demand
The #icse19 technical track has received an impressive 529 submissions! Good luck to all authors and happy reviewing to the PC and PB members :-)

Tweet übersetzen

💬 2  ⬇️ 17  ❤️ 67  ⌐
Write for the Reviewer

• Make problem statement and contribution very clear

• Use established outline: https://thesisguide.org/2014/10/13/thesis-architecture/

• Make text easily readable. This is hard and exhausting work. But you can learn it, this is no issue of talent.
All good writers start with shitty first drafts. This is how they end up with good second drafts and terrific third drafts.

— Anne Lamott

Yes, there’s hope.

First drafts don’t have to be perfect. They just have to be written.

The first draft of anything is shit.

— Ernest Hemingway

Shitty first drafts

Shitty First Drafts

The first draft of anything is shit.

— Ernest Hemingway

Shitty First Drafts

Embrace the mess
My Personal Best Practices

• Block writing time
• Begin with outline
• Separate writing from improving
• Write complete paragraphs before improving them
• Let text „cool down“ and proof-read it later again

• There is not the one silver-bullet way of writing
Scott Berkun: How to Write an Essay in Fast Motion

Youtube: http://youtu.be/BNDEDWwZyKM
English Writing Center

• Free one-to-one consulting with native English speakers
  – GR, Thesis, Homework, CV etc.
  – Text needs not to be ready

https://www.sprachenzentrum.tum.de/sprachen/englisch/english-writing-center/
Professional Lector

How to Spend Your Writing Time Well?

Every thesis comprises is made up of several chapters, such as including an introduction, definitions, related work, proposed solution, and conclusion. You must decide how much time (and pages) to spend on each of them. I call this writing resource allocation.

If this step is done poorly, authors will waste a large part of their writing time on chapters that are not central to their thesis; for example, producing bloated definitions or, a myriad of irrelevant technical details or other waste. Not only does this distract readers, it also inevitably robs authors of the time they need to write their central chapters carefully. Therefore, poor writing resource allocation is thus an effective recipe to write for a bad thesis.

So how do you do it this step well? For me, writing resource allocation is a lot like allocating plate space when eating at a large buffet. For both problems there is a similar solution strategy that is intuitive, widely applied, and reliable: to produce poor results.

To get quick results, I put pasta, and potatoes. I get more picky pickier if pork, some lamb about to leave with a full stomach of pasta, and eed to leave it out! I can't have the scallops and everybody anyone that not greedy allocation strategy.

Empty pages. To get quick results is often the introduction sections. Its can be in adding three pages just heard a course about, I say. So it should be in the

James Morrison
jmedits@gmail.com
1 Learning to Rank Extract Method Refactoring Suggestons for Long Methods

1.1 Introduction

In software engineering, refactoring has always been considered as one of the most important ways to improve software quality, thus it has received much research attention. The refactoring process can be triggered by detecting patterns which indicate code smells, or manually identifying useful refactoring opportunities. The refactoring process is generally considered as a complex, semi-automatic, and non-trivial task. To help developers identify useful refactoring opportunities, many research efforts have been made to develop refactoring tools and corresponding frameworks. However, current refactoring tools are often designed to address individual refactorings, and do not systematically handle the refactoring of long methods.

1.2 Preliminary

We make several assumptions about the refactoring learning task. In the proposed learning framework, an input feature for the refactoring learning task is an extract method refactoring suggestion for a long method.

1.3 Training and Testing

The learning process consists of two steps: training and testing. We apply random-sampling with 1000 samples, 800 for training and 200 for testing. We performed 10-fold cross-validation on the training data and used the same test set for each fold.

1.4 Evaluation

In this section, we present and evaluate the results from the learning process.

1.4.1 Research Questions

RQ1: What are the results of the learning task? Is it possible to learn a meaningful refactoring suggestion for the refactoring learning task? Is it possible to learn a meaningful refactoring suggestion for the refactoring learning task? Is it possible to learn a meaningful refactoring suggestion for the refactoring learning task?

RQ2: Can the learning process be simplified? Can the learning process be simplified? Can the learning process be simplified?

RQ3: How does the learned refactoring learning process compare with our manual refactoring suggestions? How does the learned refactoring learning process compare with our manual refactoring suggestions? How does the learned refactoring learning process compare with our manual refactoring suggestions?

RQ4: How do the extracted refactoring suggestions impact the performance of existing refactoring tools? How do the extracted refactoring suggestions impact the performance of existing refactoring tools? How do the extracted refactoring suggestions impact the performance of existing refactoring tools?

1.4.2 Results

The following paragraphs answer the research questions.

1.4.2.1 Learning Results

We applied the learning process to the refactoring suggestions for long methods. The results are shown in Table 1.

Table 1: Results of the Learning Process

<table>
<thead>
<tr>
<th>Feature</th>
<th>Accuracy</th>
<th>F1-Score</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method A</td>
<td>0.85</td>
<td>0.90</td>
<td>0.92</td>
<td>0.88</td>
</tr>
<tr>
<td>Method B</td>
<td>0.88</td>
<td>0.92</td>
<td>0.94</td>
<td>0.90</td>
</tr>
</tbody>
</table>

1.4.2.2 Refactoring Suggestions

The refactoring suggestions generated by the learning process are shown in Table 2.

Table 2: Refactoring Suggestions

<table>
<thead>
<tr>
<th>Method</th>
<th>Suggestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method A</td>
<td>Split long methods into smaller units</td>
</tr>
<tr>
<td>Method B</td>
<td>Extract helper functions</td>
</tr>
</tbody>
</table>

1.4.2.3 Impact on Existing Tools

We evaluated the impact of the extracted refactoring suggestions on the performance of existing refactoring tools. The results are shown in Table 3.

Table 3: Impact on Existing Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool A</td>
<td>15%</td>
</tr>
<tr>
<td>Tool B</td>
<td>20%</td>
</tr>
</tbody>
</table>

In conclusion, the learning process can be used to automatically generate meaningful refactoring suggestions for long methods. The extracted refactoring suggestions can improve the performance of existing refactoring tools. Future work will focus on exploring additional features and extracting more refactoring suggestions.
Prepare Presentation

https://thesisguide.org/2015/03/04/how-to-draft-your-presentation/
Presentation Differences to BA/MA

- Rehearsal talk with advisor
- Practice it in English
- Formulate starting sentences and learn them by heart
- Backup slides for questions (e.g., more details)
Forschungsarbeiten @CQSE

• Agenda: How to do a BA/GR/MA @CQSE
  – Analysis Implementation
  – Evaluation/ Case Study
  – Advisors
  – Pitch of current topics

• Remote Event in times of COVID-19

• Would like to be invited?

http://cqse.eu/feedback-tum-talk
Conclusion

Do you want to do your own research and get to know the research community? Then a guided research is the best you can do!
Thank you!

If you are interested in a guided research in the field of software analysis and testing, please let me know:

haas@cqse.eu

More Info:
www.thesisguide.org

http://cqse.eu/feedback-tum-talk