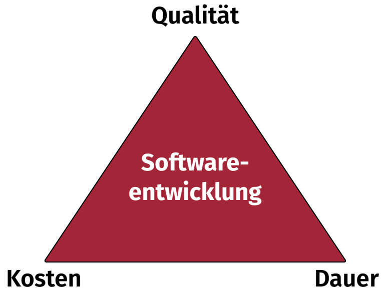
Projekte und Abschlussarbeiten

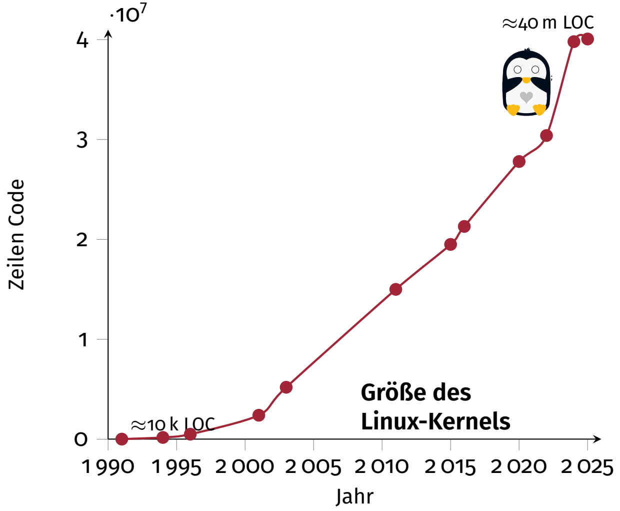
Institut für Softwaretechnik und Programmiersprachen | 3. Februar 2026

Software Engineering
Programming Languages

universität
uulm

Software Engineering





Software Engineering and Programming Languages

Prof. Tichy, Prof. Heinrich, Dr. Raschke, Dr. Wiesmayr

Domain-specific Languages

Programmier-sprachen

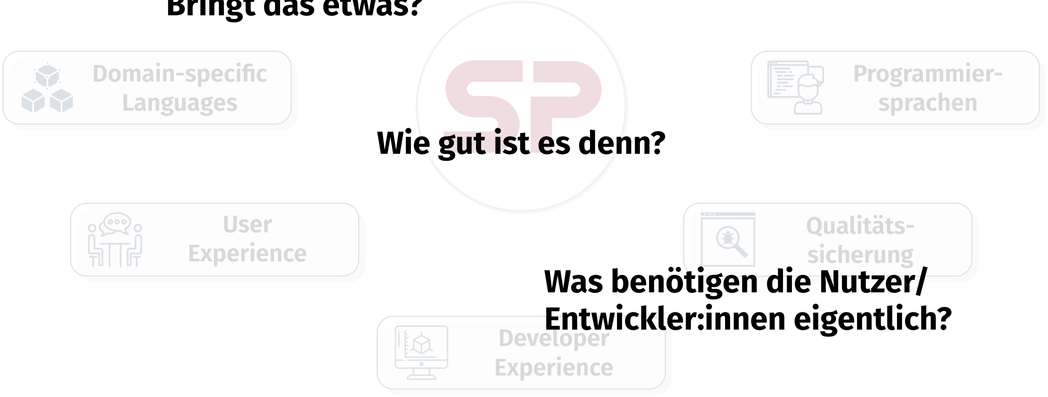
User Experience

Qualitäts-sicherung

Developer Experience

Software Engineering and Programming Languages

Prof. Tichy, Prof. Heinrich, Dr. Raschke, Dr. Wiesmayr

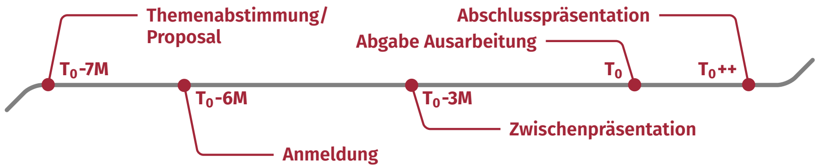


Bringt das etwas?

Wie gut ist es denn?

Was benötigen die Nutzer/ Entwickler:innen eigentlich?

Prozess



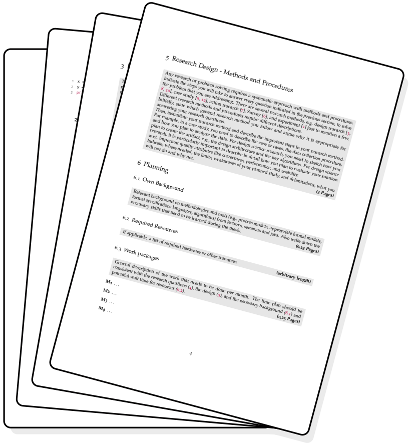
- Enge Betreuung durch Mitarbeiter:in
  - Thema
  - Methodik
  - Wöchentliche Treffen
  - Regelmäßiges Feedback
  - Kurze Wege
- Institutsbüro/Pool für Abschluss- und Projektarbeiten
- Kaffee!

Proposal

- **Ziele**
  - Abstimmung der Inhalte vor der Anmeldung
  - Risikominimierung für Sie und uns

- **Inhalte**
  - Kontext der Arbeit
  - Betrachtetes Problem / Forschungsfrage(n)
  - Stand der Forschung
  - Lösungsidee
  - Methodik und Evaluationsplan

▸ ≈ 8 Seiten Text (Reuse in der Arbeit!)



5 Research Design - Methods and Procedures

Any research or problem solving requires a systematic approach with methods and procedures. Indicate the steps you will take to answer every question indicated in the previous section, to solve the problem that you are addressing. There are several research methods, e.g. design research [3, 8, 10], case study [9, 12], action research [7], Survey [6], and experiment [1] just to mention a few. Different research methods and procedures require different descriptions. Initially, state which general research method you follow and argue why it is appropriate for answering your research questions. Then, instantiate your research method and describe the important steps in your research method. For example, in a case study, you need to describe the case or cases, the data collection procedure, and how you plan to analyze the data. For design science research, you need to sketch how you plan to create the artifact, e.g., the design architecture and the key algorithms. For design science research, it is particularly important to describe in detail how you plan to evaluate your solution w.r.t. important quality attributes like correctness, performance, and usability. Indicate, where needed, the limits, weaknesses of your planned study, and delimitations, what you will not do and why not.
(3 Pages)

6 Planning

6.1 Own Background

Relevant background on methodologies and tools (e.g., process models, appropriate formal models, formal specifications languages, algorithms) from lectures, seminars and jobs. Also write down the necessary skills that need to be learned during the thesis.
(0,25 Pages)

6.2 Required Resources

If applicable, a list of required hardware or other resources.
(arbitrary length)

6.3 Work packages

General description of the work that needs to be done per month. The time plan should be consistent with the research questions (4), the design (5), and the necessary background (6.1) and potential wait time for resources (6.2).
(0,25 Pages)

M1 …
M2 …
M3 …
M4 …

4

uulm.de/in/sp/teaching/topics-for-theses-and-projects/

Themenübersicht

1. **Softwareentwurf, -entwicklung und Usability (B. Wiesmayr)**
2. **Self-Adaptive Systems (R. Straub)**
3. **Generative AI for Control and Embedded Software Engineering (A. Raschke)**
4. **Algorithm Detection (D. Neumüller)**
5. **Static and Dynamic Program Analysis for Data Science (F. Sihler)**
6. **CodeMOrgs (A. Diera)**
7. **Architectural Security Analysis (L. Le)**

— 1 —
Softwareentwurf, -entwicklung und Usability (B. Wiesmayr)

Neu im Team: Bianca Wiesmayr

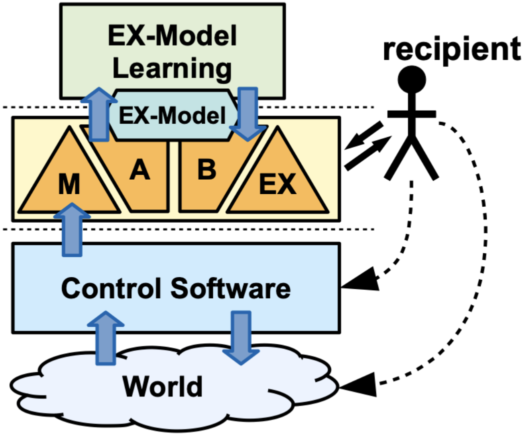
Ab 1.4.2026: Juniorprofessorin für Software Engineering

- Derzeit an der Johannes Kepler Universität Linz tätig
- 2023: Promotion in Computer Science
- **Hauptforschungsbereiche**
  - Softwareentwicklung für Automatisierungssysteme
  - KI-gestützter Entwurf von Steuerungssoftware
  - Modellgetriebene Softwareentwicklung
  - Usability von IDEs
- Open-Source-Aktivität: Weiterentwicklung des Modellierungstools Eclipse 4diac für verteilte Steuerungssysteme

— 2 —

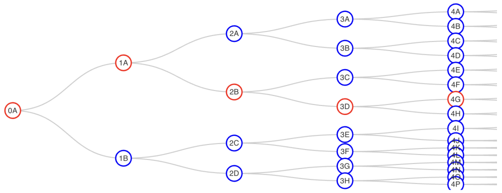
Self-Adaptive Systems (R. Straub)

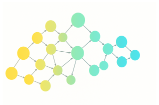
Self-Adaptive Systems

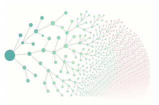


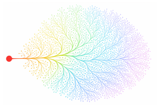
Scalable Visualization for State-Graphs

P





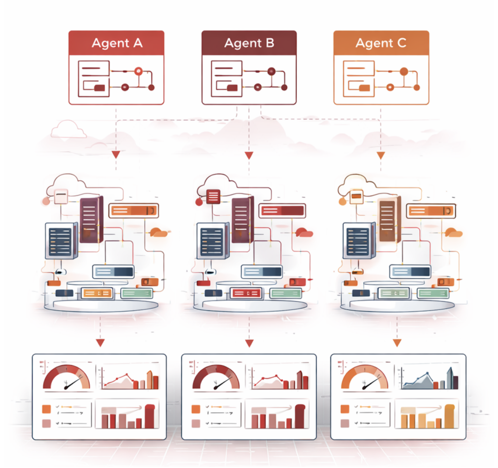




- **Problem**
  - Our state graphs get too large to display
  - We need to visualize all states in a comprehensive and understandable way
  - Ensure the visualization is intuitive and easy to understand
- **Tasks**
  - Develop visualization concepts for Large State Graphs
  - Implement the developed visualization concepts
  - Develop the solution in React/Typescript

Evaluating RL for Self-Adaptive Systems

BA



- **Idea**
  - Use Reinforcement Learning to optimize reconfiguration of cloud systems
  - Simulation based on Palladio Component Models (PCM) using Slingshot
  - Integrate Slingshot into Gymnasium environment
- **Tasks**
  - Literature Research: different RL algorithms (Pros, Cons, Setup)
  - Design a vectorized state based on simulation results **(Critical)**
  - Perform extensive experiments and evaluate different algorithms and hyperparameters

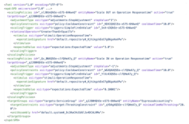
Automated Model Generation

MA

Constraints

- 4–8 Components
- < 10 SLOs
- 1–3 Policies

PCM Models



```xml
<?xml version="1.0" encoding="UTF-8"?>
<spd:SPD xmi:version="2.0" ...>
  <scalingPolicies id="_xlJ0UnQhEe-x575-6HAwnQ" entityName="Scale OUT on Operation Responsetime" active="true"
  targetGroup="_gJJ80HQhEe-x575-6HAwnQ">
    <adjustmentType xsi:type="adjustments:StepAdjustment" stepValue="1"/>
    <policyConstraints xsi:type="policy:CooldownConstraint" id="_KDXIQXQlEe-x575-6HAwnQ" cooldownTime="10.0"/>
    <scalingTrigger xsi:type="triggers:SimpleFireOnValue" id="_5nX-U3QhEe-x575-6HAwnQ"
    relationalOperator="GreaterThanOrEqualTo">
      <stimulus xsi:type="stimuli:OperationResponseTime">
        <operationSignature href="default.repository#_KjhikgzkEeCFg86gxKwVPw"/>
      </stimulus>
      <expectedValue xsi:type="expectations:ExpectedTime" value="3.0"/>
    </scalingTrigger>
  </scalingPolicies>
  <scalingPolicies id="_Qv_BNXQ5Ee-clfQHeA7y_Q" entityName="Scale IN on Operation Responsetime" active="true"
  targetGroup="_gJJ80HQhEe-x575-6HAwnQ">
    <adjustmentType xsi:type="adjustments:StepAdjustment" stepValue="-1"/>
    <policyConstraints xsi:type="policy:CooldownConstraint" id="_WGXQAXQ5Ee-clfQHeA7y_Q" cooldownTime="10.0"/>
    <scalingTrigger xsi:type="triggers:SimpleFireOnValue" id="_YlVz43Q5Ee-clfQHeA7y_Q">
      <stimulus xsi:type="stimuli:OperationResponseTime">
        <operationSignature href="default.repository#_KjhikgzkEeCFg86gxKwVPw"/>
      </stimulus>
      <expectedValue xsi:type="expectations:ExpectedTime" value="0.10001"/>
    </scalingTrigger>
  </scalingPolicies>
  <targetGroups xsi:type="targets:ServiceGroup" id="_gJJ80HQhEe-x575-6HAwnQ" entityName="EspressoAccounting">
    <targetConstraints xsi:type="target:ThrashingConstraint" id="_gYDAgXQ5Ee-clfQHeA7y_Q" minimumTimeNoThrashing="20.
    0"/>
    <unitAssembly href="default.system#_9c3KwCHcEd6lJo4DCALHMw"/>
  </targetGroups>
</spd:SPD>
```

Your Prototype

Problem:

- PCM Models are complex
- Degrees of freedom have to be identified and possible constraints defined
- An approach to solve the problem has to be implemented

Tasks:

- Analyze the PCM Models, define possible constraints, chose an approach
- Develop a Prototype
- Evaluate the Prototype

Automated Model Generation: Simple Example

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Domains
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
cID(1..8).          % Possible component IDs (max 8)
sID(1..9).          % Possible SLO IDs (max 9)
iID(1..3).          % Possible policy indices per component (up to 3)
policyType(up).     % Policy type: upscaling
policyType(down).   % Policy type: downscaling

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Component Types
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
component_type(db).
component_type(service).
component_type(cache).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Measurements (tied to component types)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
measurement(response_time, service).
measurement(availability, db).
measurement(hit_rate, cache).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Component Instances
% Exactly one type chosen for each cID. 4–8 total components overall.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1 { comp(C,T) : component_type(T) } 1 :- cID(C).

% Enforce 4–8 total components
:- #count { C : cID(C), comp(C,_) } < 4.
:- #count { C : cID(C), comp(C,_) } > 8.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Connections: undirected edges between distinct components
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
{ connected(C1,C2) } :- cID(C1;C2), comp(C1,_), comp(C2,_), C1 < C2.

% Build "edge" relation to simplify handling undirected connections
edge(A,B) :- connected(A,B).
edge(A,B) :- connected(B,A).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Graph Connectivity Constraint
% All chosen components must form a single connected component.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
reachable(X,X) :- cID(X), comp(X,_).
reachable(X,Y) :- cID(X;Y), comp(X,_), comp(Y,_), reachable(X,Z), edge(Z,

% For every pair of components (X,Y), there must be a path from X to Y
:- comp(X,_), comp(Y,_), X != Y, not reachable(X,Y).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SLOs
% Up to 9 SLOs total, each referencing a measurement (M) of some type (T)
% No two SLOs target the same measurement.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
{ slo(S,M,T) : sID(S), measurement(M,T) }.
:- #count { (S,M,T) : slo(S,M,T) } > 9.
:- slo(S1,M,T), slo(S2,M,T), S1 != S2.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Policies
% Each component must have 1–3 policies. Each policy is:
%   – Tied to component C
%   – Has an index I in 1..3
%   – Has a policy type (up/down)
%   – Targets a measurement M valid for C's type
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
{ policy(C,I,Type,M) :
    cID(C), comp(C,T), iID(I), policyType(Type), measurement(M,T) }.

% Each component must have 1–3 policies
:- comp(C,_), #count { (I,Type,M) : policy(C,I,Type,M) } < 1.
:- comp(C,_), #count { (I,Type,M) : policy(C,I,Type,M) } > 3.

% A policy must target a measurement that has a corresponding SLO
:- policy(C,I,Type,M), comp(C,T), not slo(_,M,T).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Output
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
#show comp/2.
#show connected/2.
#show slo/3.
#show policy/4.
```

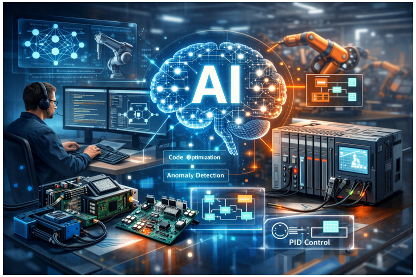
```
Answer: 1
slo(1,response_time,service) slo(9,availability,db) comp(1,service) comp(2,service) comp(3,db) comp(4,db) comp(5,db) comp(6,service) comp(7,db) comp(8,
db) policy(1,1,up,response_time) policy(1,2,up,response_time) policy(2,3,up,response_time) policy(8,2,up,availability) policy(1,2,down,response_time)
policy(6,2,down,response_time) policy(3,2,down,availability) policy(4,2,down,availability) policy(5,3,down,availability) policy(7,2,down,availability)
connected(1,2) connected(1,3) connected(2,3) connected(1,4) connected(2,4) connected(1,5) connected(2,5) connected(1,6) connected(2,6) connected(3,6)
connected(4,6) connected(5,6) connected(1,7) connected(2,7) connected(3,7) connected(4,7) connected(5,7) connected(6,7) connected(1,8) connected(2,8)
connected(3,8) connected(4,8) connected(5,8) connected(6,8)
SATISFIABLE
```

— 3 —

Generative AI for Control and Embedded Software Engineering (A. Raschke)

Generative AI for Control and Embedded Software Engineering

Overview



ChatGPT (2026), AI-generated picture.

- Code generation using LLMs is becoming increasingly common.
- However, the use of AI in the field of control and embedded software is not yet widespread
- What obstacles/problems prevent control software manufacturers from using AI?
- How can this situation be improved?

Problems and Needs in Using AI in Automation SE

MA

ChatGPT (2026), AI-generated picture.

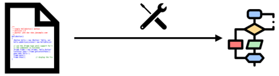
- **Problem**
  - What are the barriers for not using AI in automation SE?
  - What is necessary to improve the situation?
- **Goal**
  - Gather experiences and opinions from industry
  - List of barriers and ideas how the situation can be improved
- **Tasks**
  - Interview study with industrial partners
  - Qualitative analysis of answers
  - Coding and clustering of results
- **Needed Skills**
  - Communication skills

— 4 —

Algorithm Detection (D. Neumüller)

Algorithm Detection

Overview

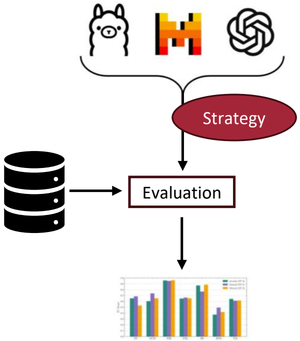

Analysis of Source Code Base
Detected algorithms and involved components

- **SW Comprehension Problems**
  - Tedious, time consuming and often manual process
  - Out of date documentation
  - Colleagues and experts not available

- **Goal**
  - Support software comprehension by automatically detectiong algorithms contained in a code base

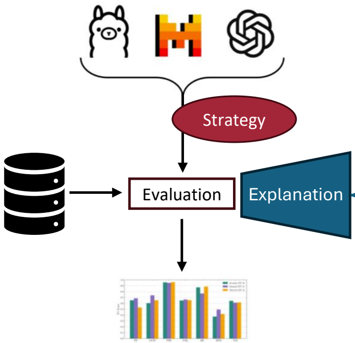
LLM Explainability in Algorithm Detection

BA MA



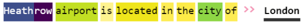
LLM Explainability in Algorithm Detection

BA MA

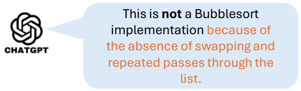


Integrated Gradients:

Which input words result in London?

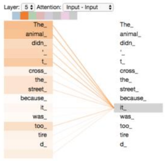


Self-Explanation:



This is **not** a Bubblesort implementation because of the absence of swapping and repeated passes through the list.

Attention-head Visualization:

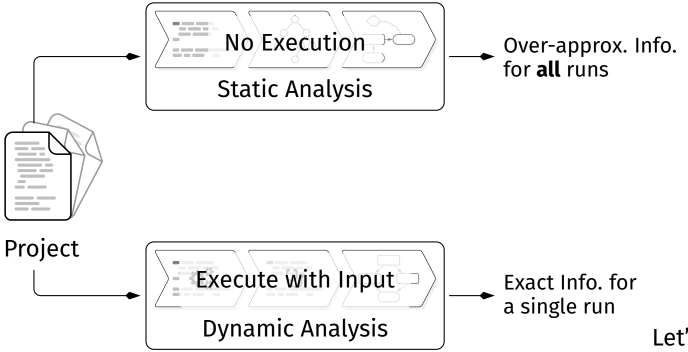


- **Goal**
  - Understand why and in which cases LLMs fail.
- **Tasks**
  - Asses different explanation techniques.
  - Select and evaluate two explanation techniques for different failure cases.
  - Possibly evaluate with different LLMs.
  - No previous ML background required.

— 5 —
Static and Dynamic Program Analysis for Data Science (F. Sihler)

Static and Dynamic Program Analysis for Data Science

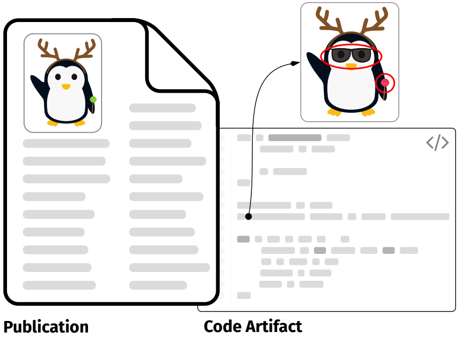
Overview



Let's dive in!

What did they do to my figure?

P BA MA



- **Problem**
  - Figures in publications may differ from those produced by the artifact
  - It is hard to link figures to code
- **Goal**
  - Identify changes in figures
- **Tasks**
  - Extract figures and compare them
  - Link figures to code via static analysis
  - Evaluate on real-world papers
- **Needed Skills**
  - Programming in a language of choice
  - Understanding of empirical eval.

Program Slicing to Improve Conventional Coverage

BA MA

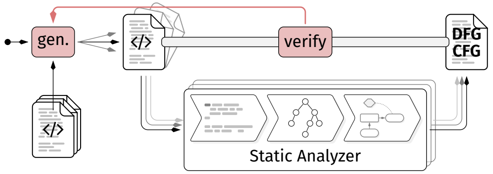
```
setup_conn ← function() {
   con ← dbConnect(SQLite(), ":memory:")
   dbExecute(con, "CREATE TABLE...")
   return(con)
}

test_that("DBI::dbExecute works", {
   expect_false(is.null(setup_conn()))
})
```

- **Problem**
  - Test coverage can be misleading
- **Goal**
  - Improve test-coverage metrics with program slicing
- **Tasks**
  - Pick a language of choice
  - Identify sensible slicing points
  - Evaluate on real-world projects
- **Needed Skills**
  - Understanding of test coverage
  - Programming in a language of choice

Fully automated Correctness Verification for Static Analysis Tools

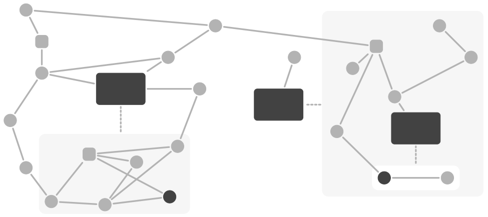
BA MA P



- **Problem**
  - Languages have many, complex and usually underspecified semantics
  - Verifying static analyses is hard
- **Goal**
  - Find automated ways to check correctness properties
- **Tasks**
  - Identify strategies like metamorphic testing or differential testing
  - Use these to generate test cases
  - Evaluate effectiveness
- **Needed Skills**
  - Knowledge of PBT or similar

Transitive Packages and Security

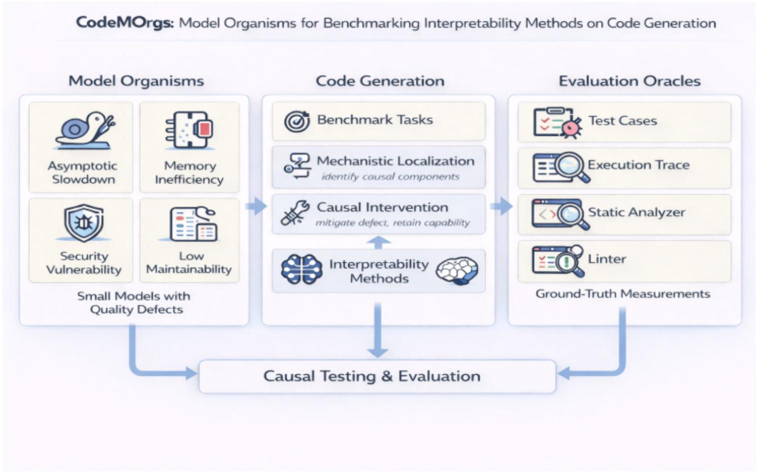
P BA MA



- **Goal**
  - Work on and extend *flowR*, a static analysis framework
- (Possible) **Tasks**
  - Security Analysis
  - Transitive Package Information
  - Dynamic Code Loading
  - Implicit Assumption Inference
  - ...
- **Needed Skills**
  - Programming in/with TypeScript

— 6 —
CodeMOrgs (A. Diera)

CodeMOrgs
Model Organisms for Benchmarking Interpretability Methods on Code Generation



Model Organisms with non-functional Code Quality Defects

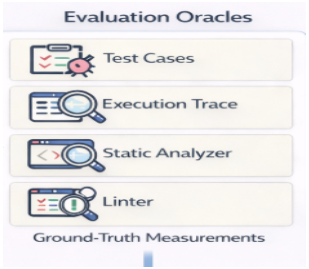
BA MA

Model Organisms
Asymptotic Slowdown
Memory Inefficiency
Security Vulnerability
Low Maintainability
Small Models with Quality Defects

- **Goal**
  - Create small models that exhibit specific coding behaviours
- **Tasks**
  - Create custom training dataset with code quality defects
  - Fine-tune small LLMs on the new dataset
  - Setup coding tasks to test the models
- **Needed Skills**
  - Python
  - Knowledge in machine learning, LLMs
  - Knowledge/Interest in at least 1 code quality aspect

Code Quality Evaluation of Large Language Models

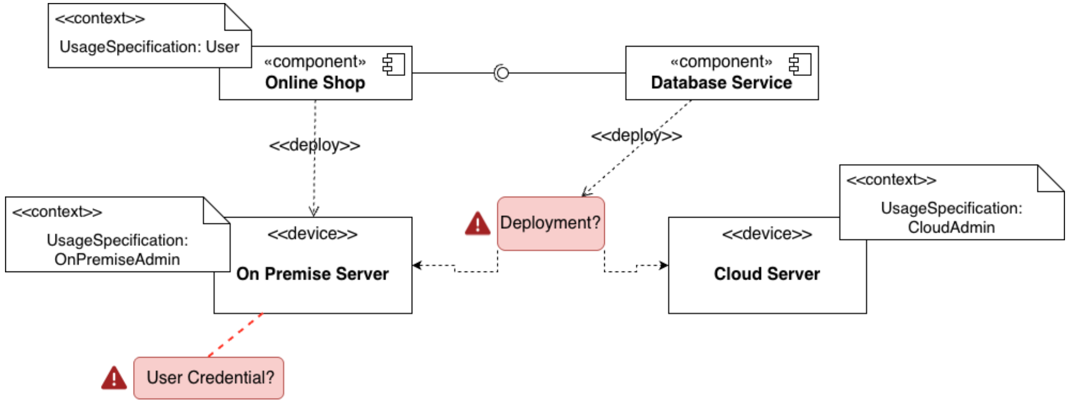
BA MA



- **Goal**
  - Create an evaluation harness for AI generated code that measures code quality
- **Tasks**
  - Define code quality metrics using formal tools
  - Develop an evaluation pipeline
  - Benchmark multiple open and closed sourced LLMs
- **Needed Skills**
  - Code Analysis tools
  - System Design
  - Knowledge/Interest in code quality

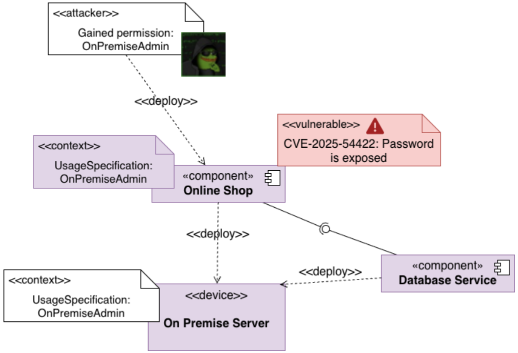
— 7 —
Architectural Security Analysis (L. Le)

Architectural Security Analysis



Mitigation of Attack Propagation using Architectural Analysis and Language Models

BA MA



- **Problem**
  - An attack can propagate and thus affect the entire cyber-physical systems.[a]
  - Selecting appropriate mitigation techniques requires a lot of expert knowledge.
  - LLM is being used in security.[b][c]

[a] Walter et al. "Architectural attack propagation analysis for identifying confidentiality issues." ICSA. IEEE, 2022.

[b] Wang et al. "Shieldgpt: An llm-based framework for ddos mitigation." Proceedings of the 8th asia-pacific workshop on networking. 2024.

[c] Gong et al. "Information security based on llm approaches: A review." arXiv preprint arXiv:2507.18215 (2025).

Mitigation of Attack Propagation using Architectural Analysis and Language Models (Cont.)

BA MA

- **Goal**
  - Using Large Language Model (LLM) to mitigate an attack propagation based on previous work.
- **Current research**
  - Architectural attack propagation analysis.[a][b]

[a] Walter et al. "Architectural attack propagation analysis for identifying confidentiality issues." ICSA. IEEE, 2022.

[b] Walter et al. "Architecture-based attack propagation and variation analysis for identifying confidentiality issues in Industry 4.0." at-Automatisierungstechnik 71.6 (2023): 443-452.

- **Overall Tasks**
  - Research and develop an approach to use architectural attack propagation analysis and LLM to mitigate an attack propagation.
  - Using architecture and asking LLM to identify the vulnerabilities and a suggestion to mitigate the vulnerabilities.
  - Analyse the attack propagation with the proposed mitigation.

Mitigation of Attack Propagation using Architectural Analysis and Language Models (Cont.)

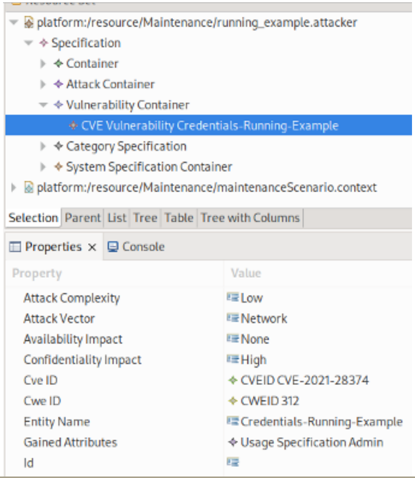
BA MA

- **Tasks in detail and the expected results**
  - Build a Retrieval-Augmented Generation (RAG) from vulnerability database[1].
  - Develop a process that allows to use a software architecture model, access control policies model as the input for LLM.
  - The output is the possible attacks and how they propagate through the architecture and the mitigation proposed by LLM.

[1] `https://nvd.nist.gov/`

Online Modelling and Analysis Tool to investigate Attack Propagation in Software Architectures

P



platform:/resource/Maintenance/running_example.attacker
- Specification
  - Container
  - Attack Container
  - Vulnerability Container
    - CVE Vulnerability Credentials-Running-Example
  - Category Specification
  - System Specification Container

platform:/resource/Maintenance/maintenanceScenario.context

Property	Value
Attack Complexity	Low
Attack Vector	Network
Availability Impact	None
Confidentiality Impact	High
Cve ID	CVEID CVE-2021-28374
Cwe ID	CWEID 312
Entity Name	Credentials-Running-Example
Gained Attributes	Usage Specification Admin
Id	

- **What we have?**
  - An attack propagation analysis tool is Eclipse-based.[a]
  - The current tool allows to model an attack and analyse its propagation.
- **BUT**
  - There is no graphical tool to model an attacker.
  - When using the tool, we need to install it into our local machine.
  - We need a tool can run on browser-based.

[a] `https://fluidtrust.github.io/attack-propagation-doc/index.html`

Online Modelling and Analysis Tool to investigate Attack Propagation in Software Architectures (Cont.)

P

- **Goal**
  - Develop an online modelling tool to allow modelling an attacker by using a website.
- **Tasks**
  - Develop an web application to support modelling an attacker.
  - Develop the function for the online tool to allow conducting an attack propagation analysis.
  - Develop the extract function of the online tool to deliver an attractive attack graph.

Online Modelling and Analysis Tool to investigate Attack Propagation in Software Architectures (Cont.)

P

- **Expected Features**
  - The new online tool should allow the architects upload their software architecture model as the input.
  - We will use Palladio Component Model [1] as the architectural description language.
  - The tool allow architects to model attacker and conduct the attack propagation analysis with the graphical view.
  - The tool should allow to download all models and import them into the Eclipse-based tool.
- **Expected Results**
  - A online modelling tool that allows users to conduct an attack propagation analysis.
  - The online tool can be shipped as a dockerized package.
- **Needed Skills**
  - Model-driven development.
  - Web application development.
  - Dockerization.

Contact

- **Who?**
  - Lan Le
  - Prof. Dr. Robert Heinrich
- **Email:**
  - lan.le@uni-ulm.de
  - robert.heinrich@uni-ulm.de

References

[1] Ralf H Reussner et al. *Modeling and simulating software architectures: The Palladio approach.* MIT Press, 2016.

uulm.de/in/sp/teaching/topics-for-theses-and-projects/

Bis bald beim Institut für Softwaretechnik und Programmiersprachen!

Vielen Dank für Ihre Aufmerksamkeit.