University of Stuttgart
Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW)

Artificial Intelligent Industrial Software Engineering

Prof. Dr.
rer. nat. habil.
Andreas
Wortmann

Enabling Domain Experts to Contributing Machine-Processable Solutions

Through better abstraction and automation

Efficient Model-Driven Software Development	Artificial Intelligence for Engineering	Methodical Model-Driven Operations
		
SCOLAR (DFG)	USE (ICM)	MBDO (DFG)
Factory-X (BMWK)	AISA (MWK)	SofDCar (BMWK)
	ML4GreenROS (ICM)	SDMflex (ICM)

Slide available from

www.wortmann.ac/presentations

McKinsey & Company

The economic potential of generative AI

The next productivity frontier

Generative AI can increase developer speed, but less so for complex tasks.

using generative AI, %

Without generative AI | With generative AI

35–45 | 20–30 | <10

Code generation | Code refactoring | High-complexity tasks

Forbes

FORBES > INNOVATION > ENTERPRISE TECH

How Generative AI Will Change The Jobs Of Computer Programmers And Software Engineers

AWS-Chef: Viele Programmierer müssen wohl bald wegen KI umsatteln

In einer geleakten Aufnahme erklärt der Cloud-Chef von Amazon, die meisten Entwickler könnten bald mit dem Programmieren dort aufhören, wo die KI übernimmt.

572

Der Alltag von Programmiereren wird sich bald ändern, meint Amazons Cloud-Chef. (Bild: Tero Vesalainen/Shutterstock.com)

25.08.2024, 13:08 Uhr Lesezeit: 3 Min.

Expertise | Insights | About | Locations | Join

Think:Act Magazine "It's time to rethink AI"

AI and the workplace

READ THE CURREN

coursera

Informatik und Ingenieurwesen > Softwareentwicklung > Will AI

roughly double if we include the

AI will liberate developer capacity: a lever towards further improving innovation

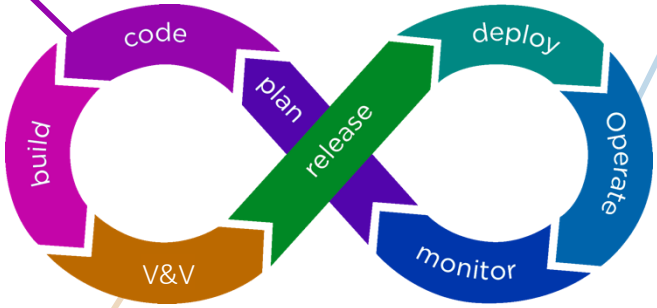
Andreas Wortmann | www.wortmann.ac | @andwor

Software Engineering is Much More Than Programming

AI for software engineering (AI4SE) can support the complete life cycle

- Generation of ST PLC code
- Agent-based LLM toolkits

- Learning energy-optimized, robot configurations



- Automated optimization of code maintainability

Andreas Wortmann | www.wortmann.ac | @andwor

PLC Code is a Cornerstone of Industrial Automation

Liberation of innovation capacity through LLMs

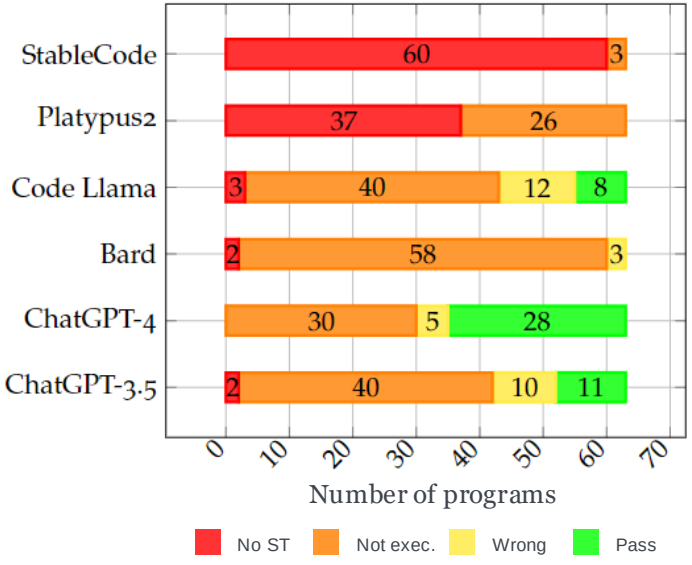
Code Completion and Suggestions	Real-time suggestions and code completion options, reducing the time engineers spend writing and debugging PLC code.[1]
Error Detection and Correction	Identify and suggest corrections for errors in PLC code, enhancing the reliability of industrial automation systems.
Optimization of Code for Efficiency	Analyze existing code for inefficiencies and recommend optimizations, leading to more streamlined and cost-effective operations.[2]

[1] Koziolek, H., Grüner, S., Ashiwal, V.: ChatGPTfor PLC/DCS Control Logic Generation. ETFA 2023.
[2] Koziolek, H., Grüner, S., Hark, R., Ashiwal, V., Linsbauer, S., & Eskandani, N.: LLM-based and Retrieval-Augmented Control Code Generation. LLM4Code 2024.

But the LLMs it Needs Specific Instructions, Training, Data

Observations on the use of general-purpose LLMs for generating structured text[1]

- LLMs included: ChatGPT 3.5 & 4, Google Bard (now Gemini), Code Llama 7B, Platypus 2 13B, StableCode 3B
- Prompts implying different language constructs from various sources (exercises, websites, OSCAT[2])
- Different processes, math. functions, function blocks, …
- Evaluation: prompt ⇒
  - CodeBERT: similarity to reference solution (67%-80%)
  - pass@*k* score: top *k* programs pass test (*k*=3: 0%-84%)
- Syntactic errors with control structures, variable declarations, return statements, string handling

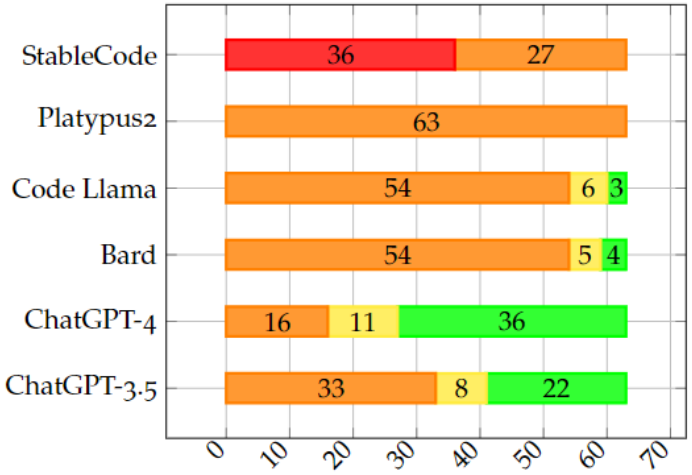


[1] Jérôme Pfeiffer, Jingxi Zhang, Kilian Tran, Andreas Wortmann, Bianca Wiesmayr. Generating PLC Code with Universal Large Language Models. ETFA 2024.
[2] OSCAT Library: http://oscat.de/de/

LLMs Tend Towards the Statistical Mean (= Mainsteam) Answers

Especially prompts need to be precise and consider this

```
You are an IEC-61131-3 Structured text coding
assistant. Your task is to generate Structured Text
with the functionality that I provide. You should
only output Structured Text. Describing text should
be minimal.
Follow these rules:
– declare variables between VAR and END_VAR
– always end control structures with END_<control
  structure>;
– avoid switch-case statements unless instructed in
  the task
– use the right quotation marks for strings
– when implementing a function: declare variables
  within the function; store return value in a
  variable that is named after the function;
– use RETURN correctly: returns to the main method,
  does not return a value
```



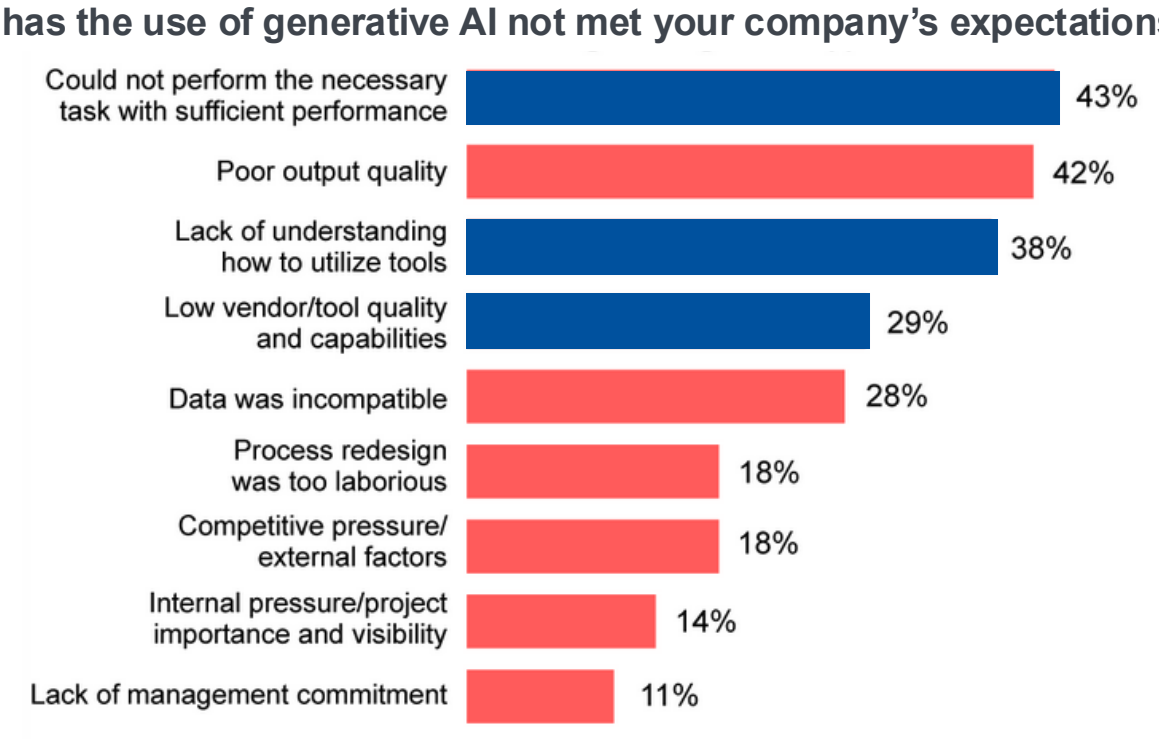
Modell	CodeBERTScore	pass@1	pass@2	pass@3
Bard	0.86721	6,35%	12,39%	18,14%
ChatGPT-3.5	0.86954	34,92%	58,01%	73,16%
ChatGPT-4	0.86010	57,14%	82,03%	92,63%
Code Llama	0.86171	4,76%	9,27%	13,82%

Effort spent in prompting can pay off immediately, despite erratic LLM performance

[1] Jérôme Pfeiffer, Jingxi Zhang, Kilian Tran, Andreas Wortmann, Bianca Wiesmayr. Generating PLC Code with Universal Large Language Models. ETFA 2024.

Poor Fit Between Requirements and Solutions Hamper AI Adoption[1]

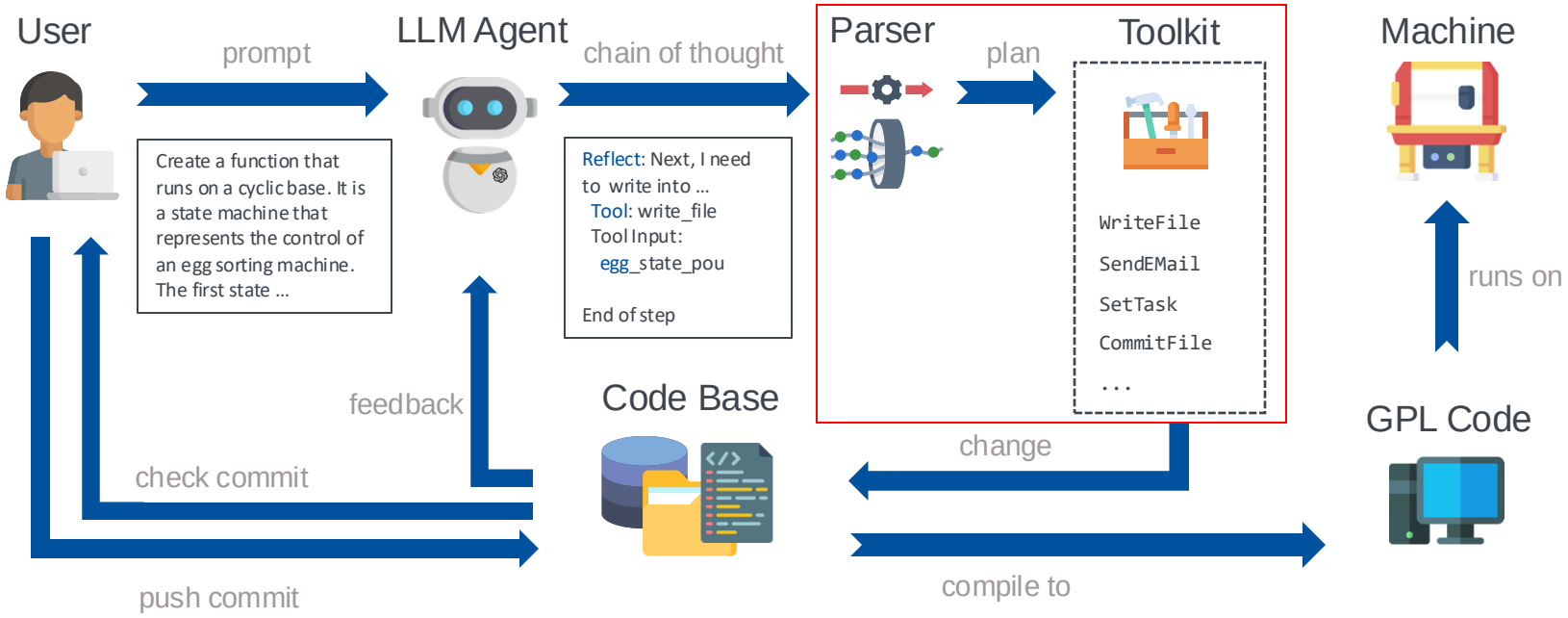
Low task performance, poor tool utilization, incompatible data, process mismatch



[1] BAIN & COMPANY: https://www.bain.com/insights/ai-survey-four-themes-emerging/ (June 2024)

Company-Specific Software Engineering Toolkits Based on LLMs

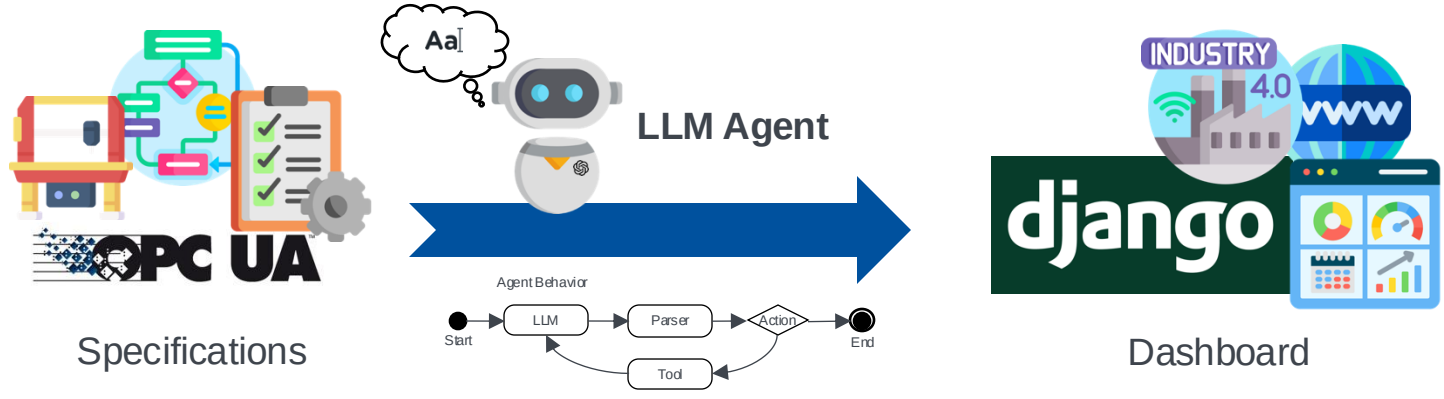
Mitigate worker shortage, solve standard tasks, accelerate innovation



Andreas Wortmann | www.wortmann.ac | @andwor

Company-Specific Software Engineering Toolkits Based on LLMs

Case study: dynamic web applications from natural language



Specifications

Dashboard

Results

- LLM agent-based design and implementation
- Creating of functional apps from different specifications

Solved

- True no-code solution
- No output restriction
- Modular architecture

Future Work

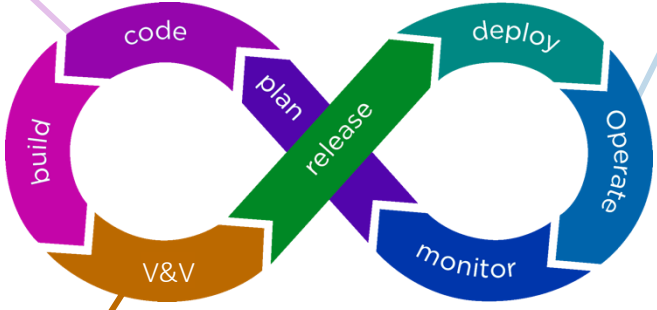
- Frontends not production-ready
- Liability/quality control unclear
- Handling of high complexity

Software Engineering is Much More Than Programming

AI for software engineering (AI4SE) can support the complete life cycle

- Generation of ST PLC code
- Agent-based LLM toolkits

- Learning energy-optimized, robot configurations

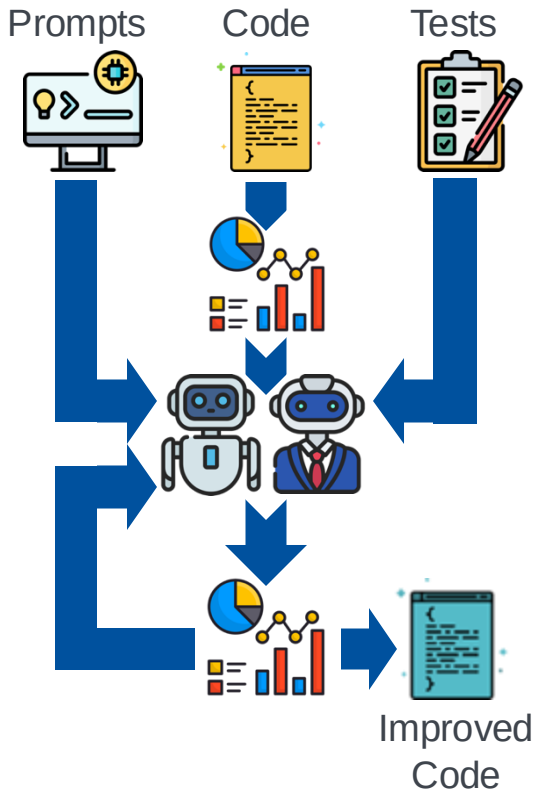


- Automated optimization of code maintainability

Code is Read more Often than Written

Making code better understandable takes time and effort

- Effort better spent on innovating; but
  hardly understandable code = hardly maintainable code
- Relevant for developing software
  - in larger teams of diverse expertise
  - that needs to be maintained for years
- Let LLMs help us to improve code by automated refactoring towards reducing code complexity
- Approach uses 2 software agents
  - editor: makes (restricted) changes to code base
  - reviewer: runs tests and evaluates improvement
- Optimization against maintainability index



Measuring Understandibilty by the Proxy of Complexity

Lines of code, cyclomatic complexity, Halstead metrics

Raw Metrics (program)

- logical lines of code (LLoC)
- cyclomatic complexity (CC)
- number of comments

Halstead (language & program): counts…

- distinct operators (n_1): ++, if , return
- distinct operands (n_2): variables and constants
- total number of operators (N_1) & operands (N_2)

→ comprehension difficulty: D = $\frac{n_1}{2} * \frac{N_2}{n_2}$

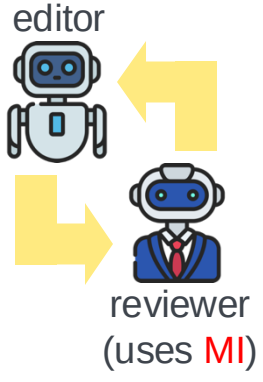
→ Halstead Volume: $(N_1 + N_2) * \log_2(n_1 + n_2)$

Maintainability Index (MI, 0 - 100)

- MI = 171 - 5.2 * *ln*(HV)
  - 0.23 * CC
  - 16.2 * *ln*(LLoC)
  + 50 * *sin*(√2.46 * rad(comments))

Optimized via

- ↓ Halstead Volume
- ↓ Cyclomatic Complexity
- ↓ Lines of Code
- ↑ Comments



Maintainability Optimization can be Fully Automated

Works best for mainstream code with good test coverage (and ChatGPT 4)

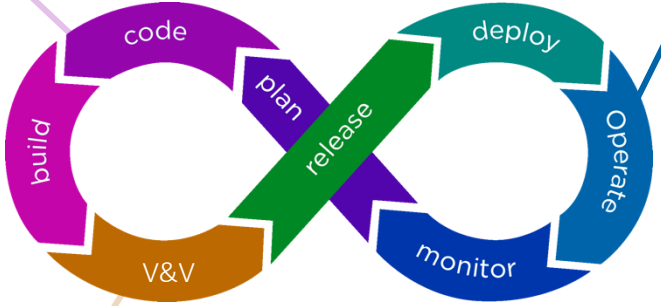
Local changes lead to better maintainable code through for almost free

Software Engineering is Much More Than Programming

AI for software engineering (AI4SE) can support the complete life cycle

- Generation of ST PLC code
- Agent-based LLM toolkits

- Learning energy-optimized, robot configurations

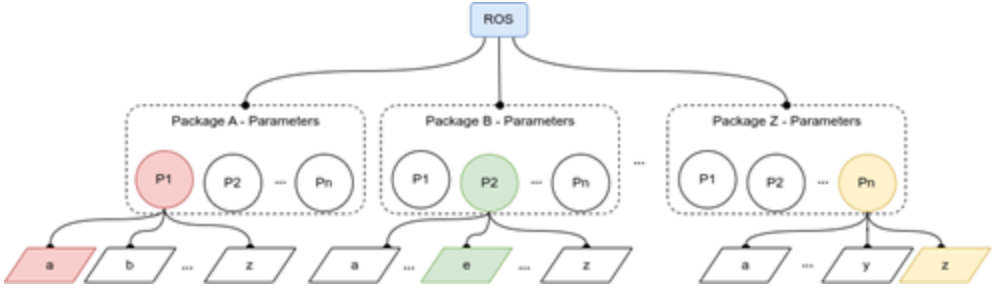


- Automated optimization of code maintainability

Many fixed Robot Programs Waste Resources

ROS nodes are highly configurable resulting in 1000s of possibilities

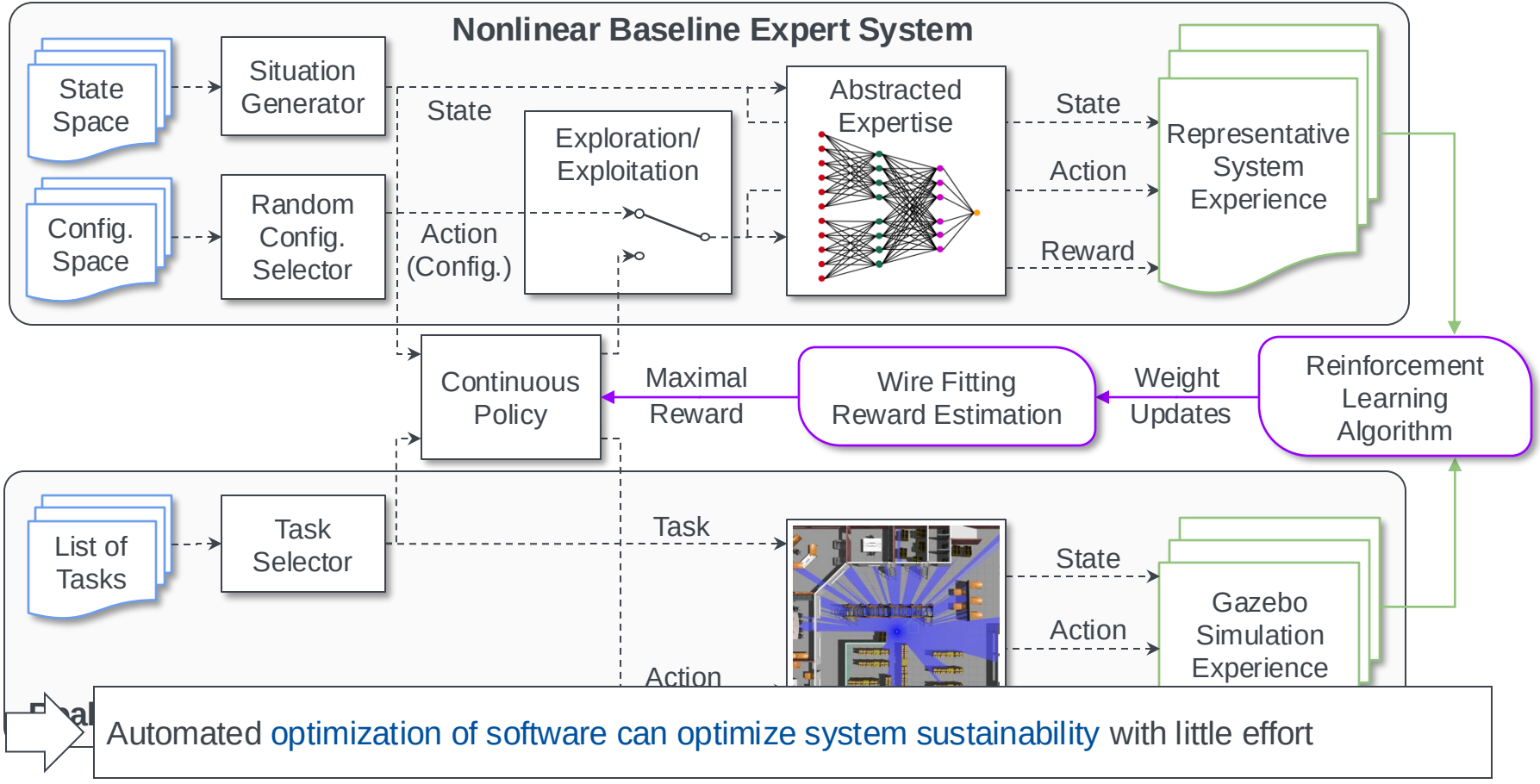
- Fixed ROS configurations tend to be sub/superoptimal for dynamic environments
  - too bad ⇒ performance problems (e.g., computation takes too long)
  - too good ⇒ allocate more resources than necessary (waste of resources)



- Goal: reconfigure ROS nodes at runtime according to the scenario dynamism
- Requires technique for selecting pareto-optimal configurations for the given scenario/mission state

[1] Wete, E., Greenyer, J., Wortmann, A., Kudenko, D., & Nejdl, W. MDE and Learning for flexible Planning and optimized Execution of Multi-Robot Choreographies. ETFA *2023.*
[2] Wete, E., Greenyer, J., Wortmann, A., Flegel, O., & Klein, M. Monte carlo tree search and GR (1) synthesis for robot tasks planning in automotive production lines. MODELS 2021.

Continuously Learning Better Robot Configurations



Andreas Wortmann | www.wortmann.ac | @andwor

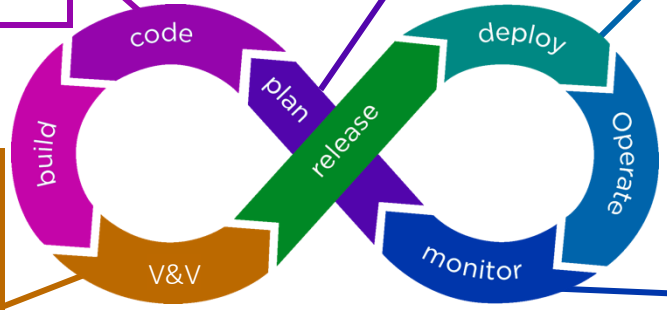
Outlook: More and More Specialized AI-Support for All Phases of the SDLC

From plan over build, release to deploy, and monitor

- Task assignment based on personality type classification[1]
- Automated sprint planning[2]

- Natural-language based code reuse[5]

- Correct-by-construction robot collaboration program synthesis with AI and formal methods[7]



- Deep-learning based prediction of test flakiness[3]
- Test-case generation with LLMs[4]

- Explaining Cyber-Physical System Behavior with Intelligent Digital Twins[6]

[1] Tunio, M. Z, et al. (2018). Crowdsourcing Software Development: Task Assignment Using PDDL Artificial Intelligence Planning
[2] Dam, H. K., Tran, T., Grundy, J., Ghose, A., & Kamei, Y. (2019,). Towards effective AI-powered agile project management
[3] Fatima, S., Ghaleb, T. A., & Briand, L. (2022). Flakify: A black-box, language model-based predictor for flaky tests.
[4] Wang, J., Huang, Y., Chen, C., Liu, Z., Wang, S., & Wang, Q. (2023). Software testing with large language model: Survey, landscape, and vision.
[5] Bader, J., et al. (2021). AI in software engineering at Facebook
[6] Michael, J. Schwammberger, M. , Wortmann, A. (2023). Explaining Cyber-Physical System Behavior with Digital Twins.
[7] Kirchhof, J. C., Kleiss, A., Rumpe, B., Schmalzing, D., Schneider, P., Wortmann, A. (2022). Model-driven self-adaptive deployment of internet of things applications with automated modification proposals.

AI-Based Software Engineering Assistants are There to Stay

And they impact competitiveness already

AI Dev Codes	ChatGPT Plus	ExplainDev	Krater.ai	SinCode AI
AI Query	Cmd J	Fig	Lightly	Snappify
AirOps AI Data Sidekick	Codacy	Figstack	Marve Chat	SourceAI
AIWriter	Code GPT	Ghostwriter	MutableAI	Sourcegraph Cody
aiXcoder	Code Snippets AI	GitFluence	Noya	SpellBox
AlphaCode	CodeAssist	GitGab	OpenAI Codex	StarCoder
Amazon CodeWhisperer	Codefy.ai	GitHub Copilot	Phind	Stenography
AskCodi	CodeGeeX	GPT95	Programming Helper	Tabnine
Autocode	CodeGen	Hacker AI	ProMindGPT	Vivid
Bard	Codeium	IntelliCode	Q	Warp AI
Bito	CodeSquire	IntelliSense	Raycast	What The Diff
Blackbox AI	CodeWP	Jedi	Refact	Wing Python IDE
Bloop	CometCore	JetBrains Datalore	Refraction AI	YouChat
BotCity	Denigma	K.Explorer	Replit	Zentask
Buildt	DevBox	Kite	Safurai	
ChatGPT	DevKit	Kodezi	Second	

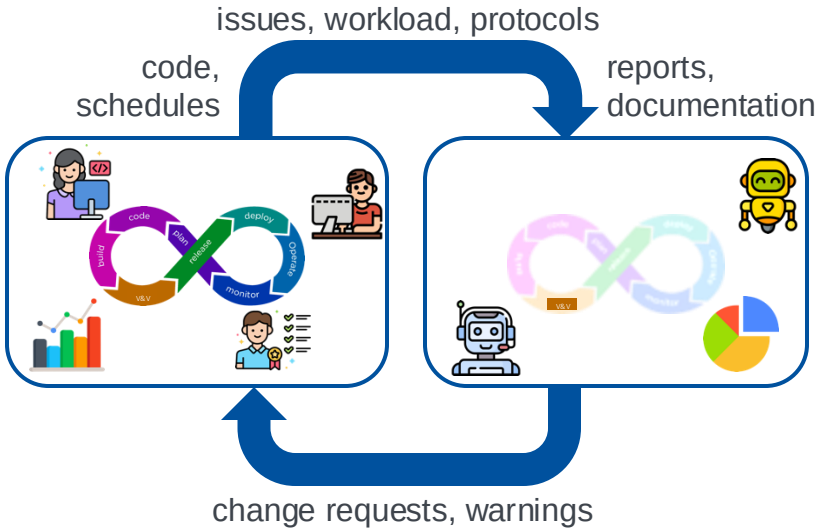
Prof. Dr. Ralf Lämmel
University of Koblenz
https://youtu.be/l9ZpURBzAwY
(July 2023)

Automating Software Engineering with Digital Twins

Twins without physical counter parts (cf. Facebook's cyber-cyber twins[1])

- Twin the software engineering process from plan to monitor
- By observing the engineering trough issues, merge requests, live code changes
- And act on the process through, for example, …

Activity	Input	Reasoning	Output
Code	Version change, local code patterns	Check git for change pull requests	Change sugges-tions for local code patterns
V&V	Editing	Compare code w. architecture	Architectural drift warnings
Monitor	Bug reports	Identify relations	Prioritize issues

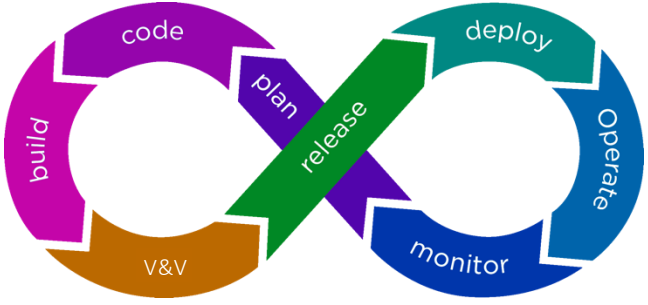


[1] Ahlgren, J., Bojarczuk, K., Drossopoulou, S., Dvortsova, I., George, J., Gucevska, N., ... & Zhou, N. (2021). Facebook's cyber–cyber and cyber–physical digital twins. In Proceedings of the 25th International Conference on Evaluation and Assessment in Software Engineering.

The Future of Industrial Software Engineering will be Exciting

AI makes planning, creating, testing, deploying, operating SW more efficient

- We will need less code craftsmen and more orchestrators: evaluate & integrate AI-generated artifacts
- Understanding SE artifacts in their context increasingly important (e.g., ISO 25010)
- AI needs the right context: train your own tools, prompt, give access to local data (e.g., RAG)
- Highly-specific and contextual knowledge important: most LLMs subpar at non-mainstream GPLs
- Prompts are important SE artifacts themselves: manage prompts FAIRly (findable, accessible, interoperable, reusable)



Highly-specialized AI tools will be your co-pilots across the complete SDLC: leverage them

University of Stuttgart
Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW)

Prof. Dr. rer. nat. habil. Andreas Wortmann

email wortmann@isw.uni-stuttgart.de

web www.wortmann.ac

phone +49 (0) 711 685-84624

twitter @andwor

University of Stuttgart
Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW)
Seidenstrasse 36 • 70174 Stuttgart • Germany

Image Attribution

- Metric icons created by Design Circle - Flaticon
- Checklist icons created by Freepik - Flaticon
- Cmd icons created by Freepik - Flaticon
- Source code icons created by Flat Icons - Flaticon
- Robot icons created by Freepik - Flaticon
- Virtual-assistant icons created by juicy_fish - Flaticon
- Robot icons created by juicy_fish – Flaticon
- Developer icons created by Paul J. - Flaticon
- Soft skills icons created by Flat Icons - Flaticon
- Computer icons created by monkik - Flaticon
- Graph icons created by Freepik - Flaticon
- Bar graph icons created by Freepik - Flaticon
- Robot icons created by Smashicons - Flaticon