TECHNISCHE UNIVERSITÄT BERLIN

The potential of AI in next-generation test automation

Prof. Dr.-Ing. Ina Schieferdecker | TAV 50, Kaiserslautern, Nov 14, 2024

Congrats TAV! Congrats TAV Community!

TAV 1, Benthe near Hannover, 1991

Softwaretechnik-Trends , Band 11, Heft 2, S. 53-84

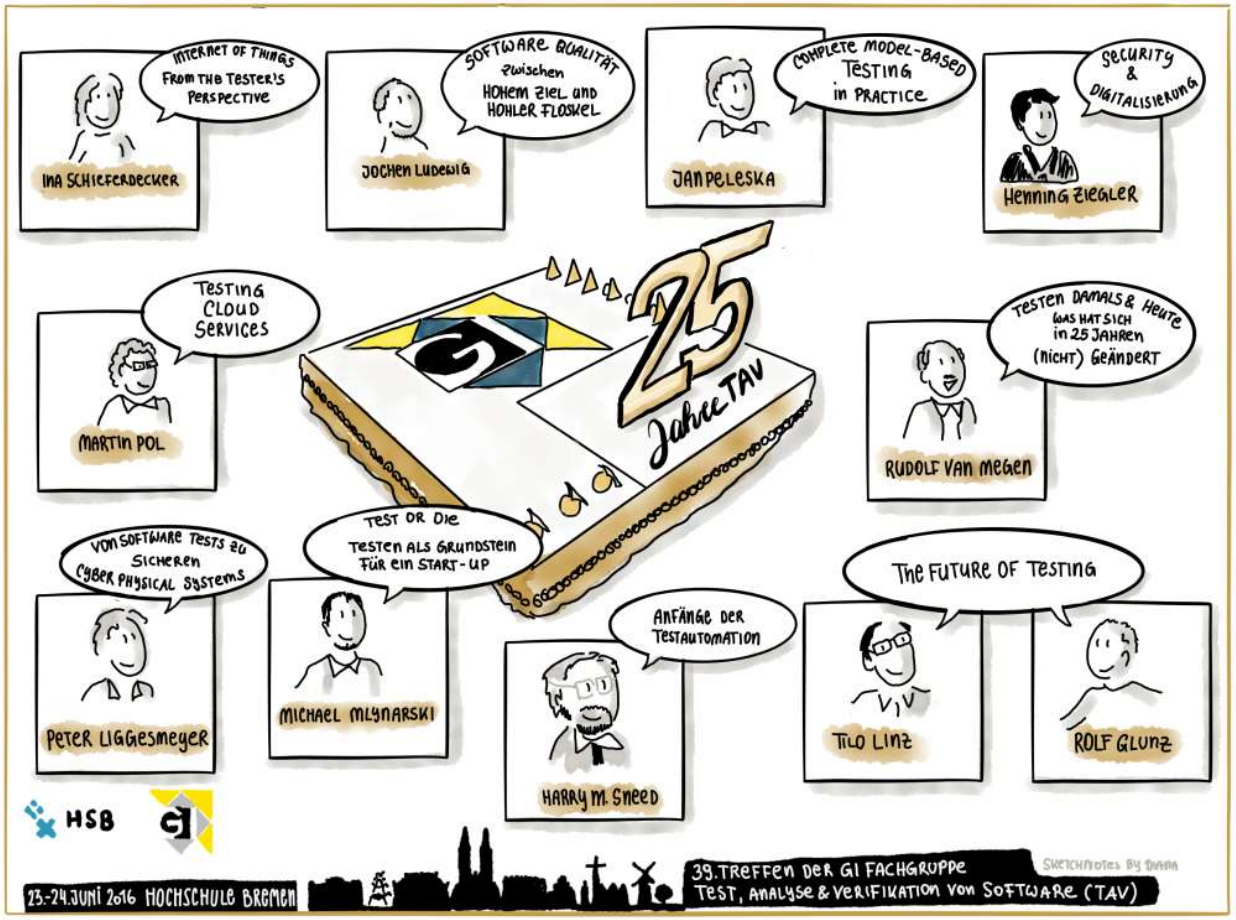
- P. Liggesmeyer, Ruhr-Universität Bochum:
  Klassifikation der Verfahren zur analytischen Software-Qualitätssicherung auf der Modulebene
- K. Grimm, Daimler-Benz:
  Kriterien zur Klassifizierung von Methoden und Verfahren zum Software-Test
- E. H. Riedemann, Universität Dortmund:
  Starke Mutationsanalyse und symbolische Ausführung als Mittel zur fehlerorientierten Testdatenerzeugung
- M. Müllerburg, GMD:
  Software Test: Methode und Entwicklungsphase
- A. Spillner, Universität Bremen:
  Testmethoden und Testdatengewinnung für den Integrationstest modularer Softwaresysteme
- U. Buhrandt, U. Krutschinna, SEL:
  Test eines komplexen Softwaresystems
- H.-J. Kreowski, Universität Bremen:
  Testen modular strukturierter Systeme aus algebraischer Sicht
- H. Schippers, RW-TÜV:
  Ein Ansatz für den Test nebenläufig ausführbarer modularer Systeme auf der Basis formaler Spezifikation
- R. van Megen, SQS:
  Systematisches Testen
- U. Pelkmann, SCOPE:
  Software-Test: Erfahrungen aus der Praxis, Stand der Entwicklung von TESTSCOPE
- W. Lamprecht, Siemens:
  COMTESS
- H. Sneed, SES:
  PC-TEST - Die integrierte Testumgebung
- D. Gschwender, Siemens Nixdorf Informationssysteme:
  Das SINIX-Funktionstest-Environment SFTE - ein Werkzeug zur automatischen Qualitätssicherung
- L. Eichler, GMD:

Congrats TAV! Congrats TAV Community!

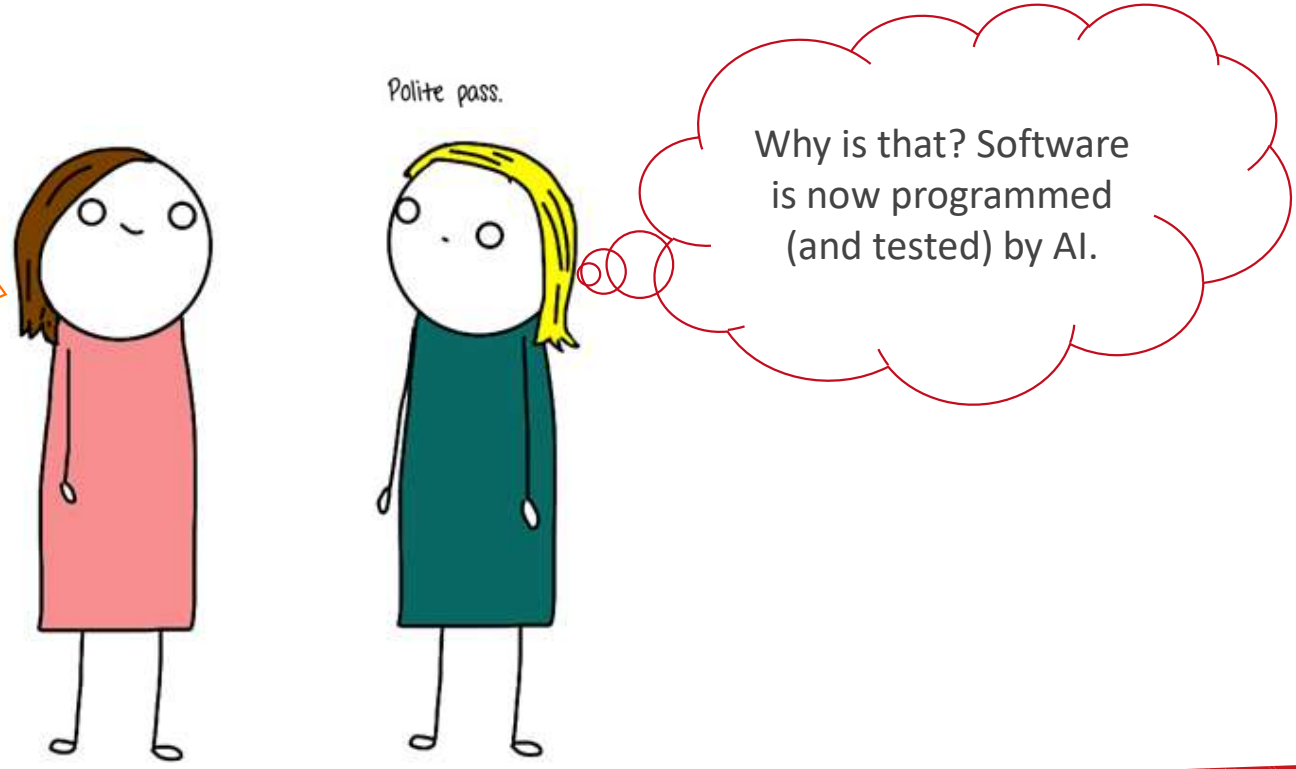
TAV 1, Hannover, 1991

25 Years of TAV, TAV 39, Bremen, 2016

50 Years of TAV, TAV 78, ?, 2041 (?)



I am currently getting to know current programming frameworks with great pleasure and am enthusiastic about the possibilities with modern programming of data analyses and AI.



ChatGPT 4: Comparison of Human and (Mimiced) Artificial Intelligence

Aspect	Humans …	AI-based systems …
Learning Mechanism	… learn through experience, education, and social interaction.	… learn through data analysis, training algorithms, and feedback loops.
Creativity	… can exhibit creativity, imagination, and emotional intelligence.	… lack true creativity and emotions, but can simulate creative behaviors based on predefined rules or algorithms.
Adaptability	… can adapt to new situations, environments, and challenges.	… can adapt within the parameters defined by their algorithms or training data, but may struggle with unexpected scenarios.
Error Tolerance	… can tolerate and learn from mistakes, errors, and uncertainties.	… may fail or produce inaccurate results when faced with novel or ambiguous situations, unless specifically designed to handle such scenarios.
Contextual Understanding	… have a deep understanding of context, nuance, and cultural factors.	… may struggle with understanding context, sarcasm, irony, or cultural nuances, unless explicitly programmed to recognize and interpret them.
Consciousness	… possess consciousness, self-awareness, and subjective experiences.	… lack consciousness and subjective experiences; it operates solely based on algorithms and data processing.
Scalability	Human intelligence is limited by factors such as time, physical constraints, and cognitive capacity.	… can scale to process and analyze vast amounts of data much faster and more efficiently than humans.
Ethical Considerations	… possess moral agency and can make ethical decisions based on empathy, values, and social norms.	… require ethical guidelines and oversight to ensure responsible use and mitigate potential biases or unintended consequences.

ChatGPT 4: Human capabilities important for software testing

Capability	Importance in Software Testing for …
Analytical Thinking	… analyzing requirements and designing test cases, as well as understanding the system's complexities.
Attention to Detail	… spotting discrepancies, errors, and subtle bugs.
Creativity	… imagining potential use cases and finding unexpected bugs.
Problem-Solving Skills	… troubleshooting issues and finding solutions to defects.
Critical Thinking	… evaluating the implications of test results and questioning assumptions.
Communication Skills	… documenting bugs, explaining reproduction steps, and relaying findings.
Patience and Perseverance	… enduring repetitive tasks and long bug-finding processes.
Adaptability	… quickly learning new tools and adapting to changes in technology and requirements.
Empathy and User Perspective	… ensuring the software meets user needs and is user-friendly.
Technical Knowledge	… understanding software architecture and code specifics.
Teamwork	… collaborating with other team members and aligning with project goals.
Time Management	… managing multiple tasks and meeting project deadlines.

ChatGPT 4: AI be able to test AI itself?

- In theory, it's possible for AI to test AI systems without human intervention, however, there are several challenges and limitations:
  - Complexity of AI Systems
  - Interpretability and Explainability
  - Biases and Blind Spots
  - Adaptability to Novel Scenarios
  - Ethical Considerations
- AI-based testing tools and techniques can augment human testers' capabilities and automate certain aspects of the testing process
- Human judgment, creativity, and domain expertise remain essential for ensuring the quality, reliability, and ethical integrity of AI systems.
- While AI may eventually play a significant role in testing AI systems, human involvement is likely to remain necessary **for the foreseeable future**.

AI Developments

Sep 17, 2024: Sopra Steria Next

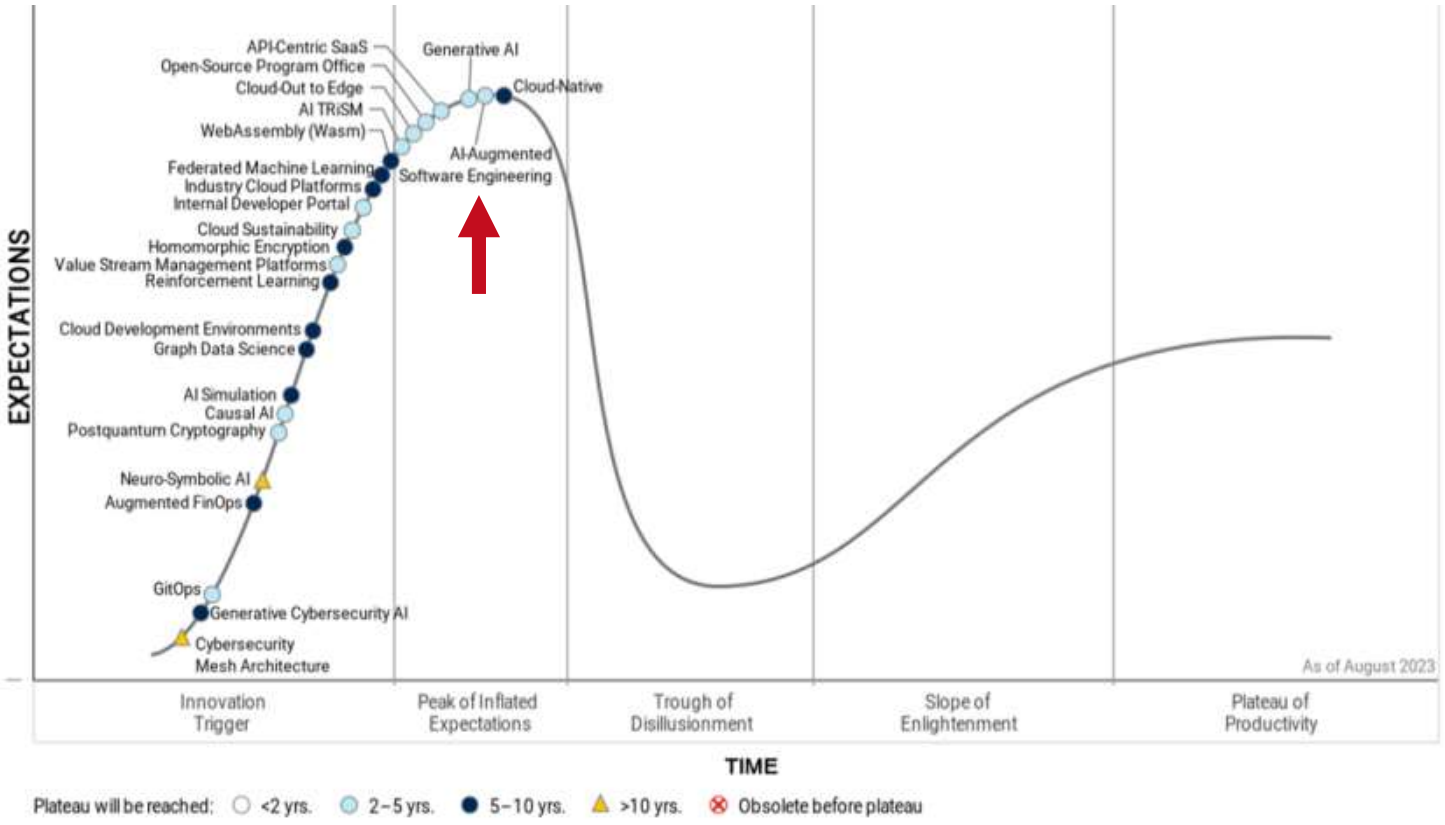
- Market analysis based on four AI categories: AI for Machines, Processes, Humans, Software
- Predicts global AI market will double to $1,270 billion by 2028 with an annual increase of 19 %

AI for *Software*

- Greatest potential together with AI for People
- Developing primarily in the financial services, healthcare, e-commerce and media domain
- Supporting not only programmers, but enabling even non-experts to write complex software

https://www.soprasteria.co.uk/insights/press-releases/details/sopra-steria-next-predicts-ai-market-will-double-to-1-270-billion-by-2028

AI-Augmented Software Engineering, 2023

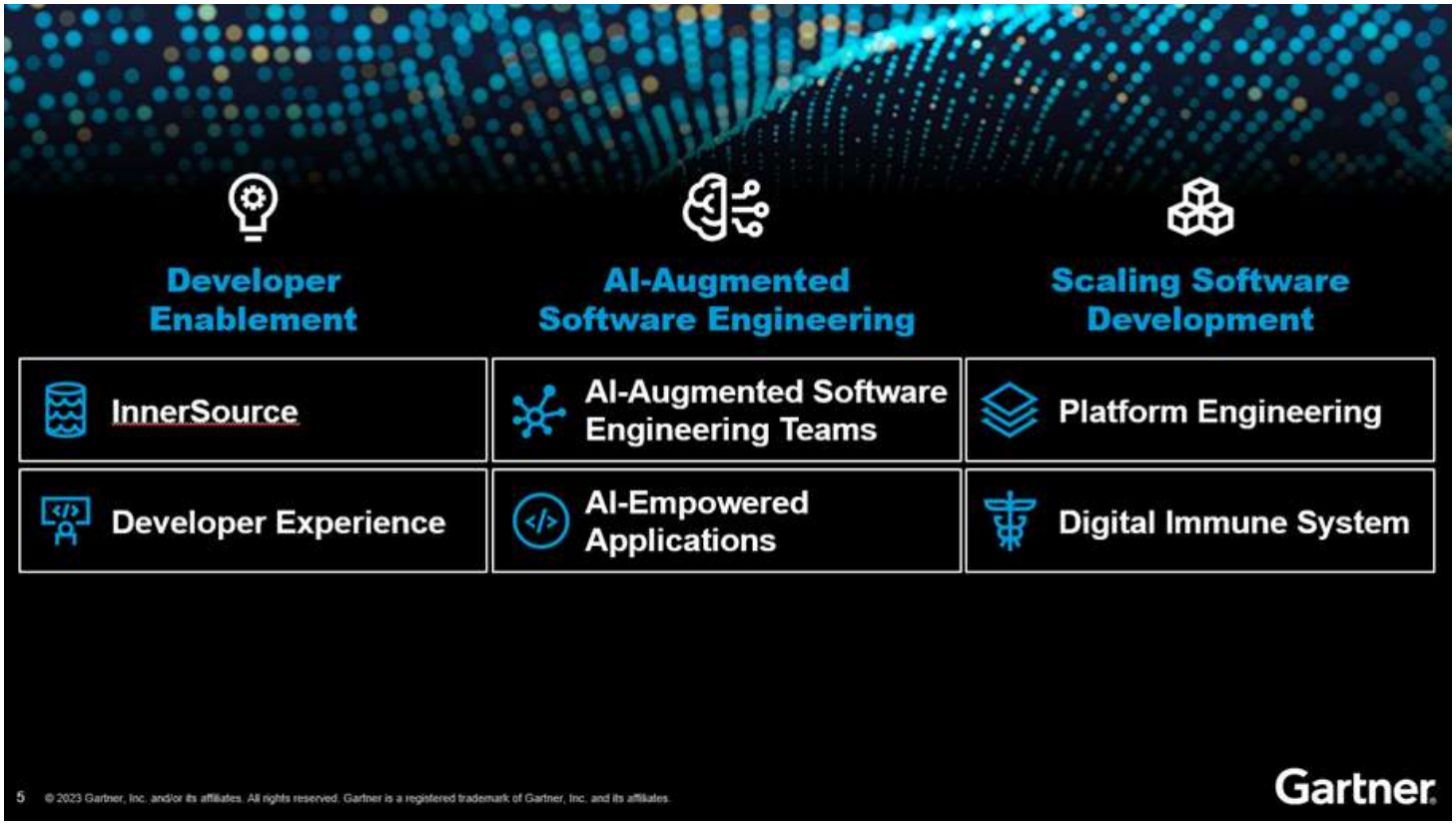


https://www.gartner.com/en/newsroom/press-releases/2023-11-28-gartner-hype-cycle-shows-ai-practices-and-platform-engineering-will-reach-mainstream-adoption-in-software-engineering-in-two-to-five-years

AI practices and platform engineering will reach mainstream adoption in software engineering in two to five years

By 2027, 50% of enterprise software engineers will use machine learning-powered coding tools

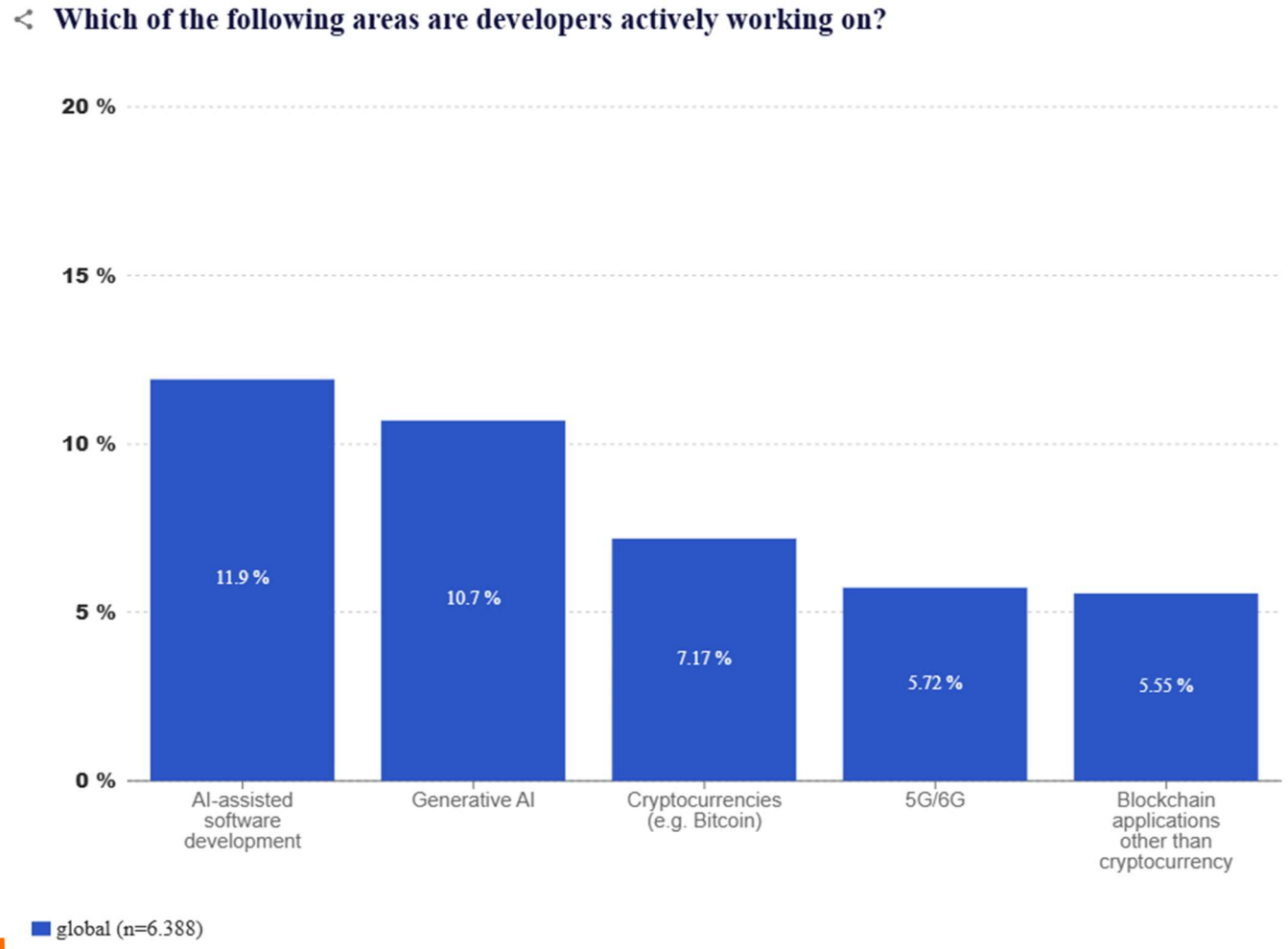
Top Strategic Technology Trends for Software Engineering, 2023



https://www.gartner.com/en/newsroom/press-releases/gartner-identifies-the-top-strategic-technology-trends-in-software-engineering-trends-for-2023

Top trends … that software engineering leaders must leverage over the next two to three years to stay ahead of the curve …

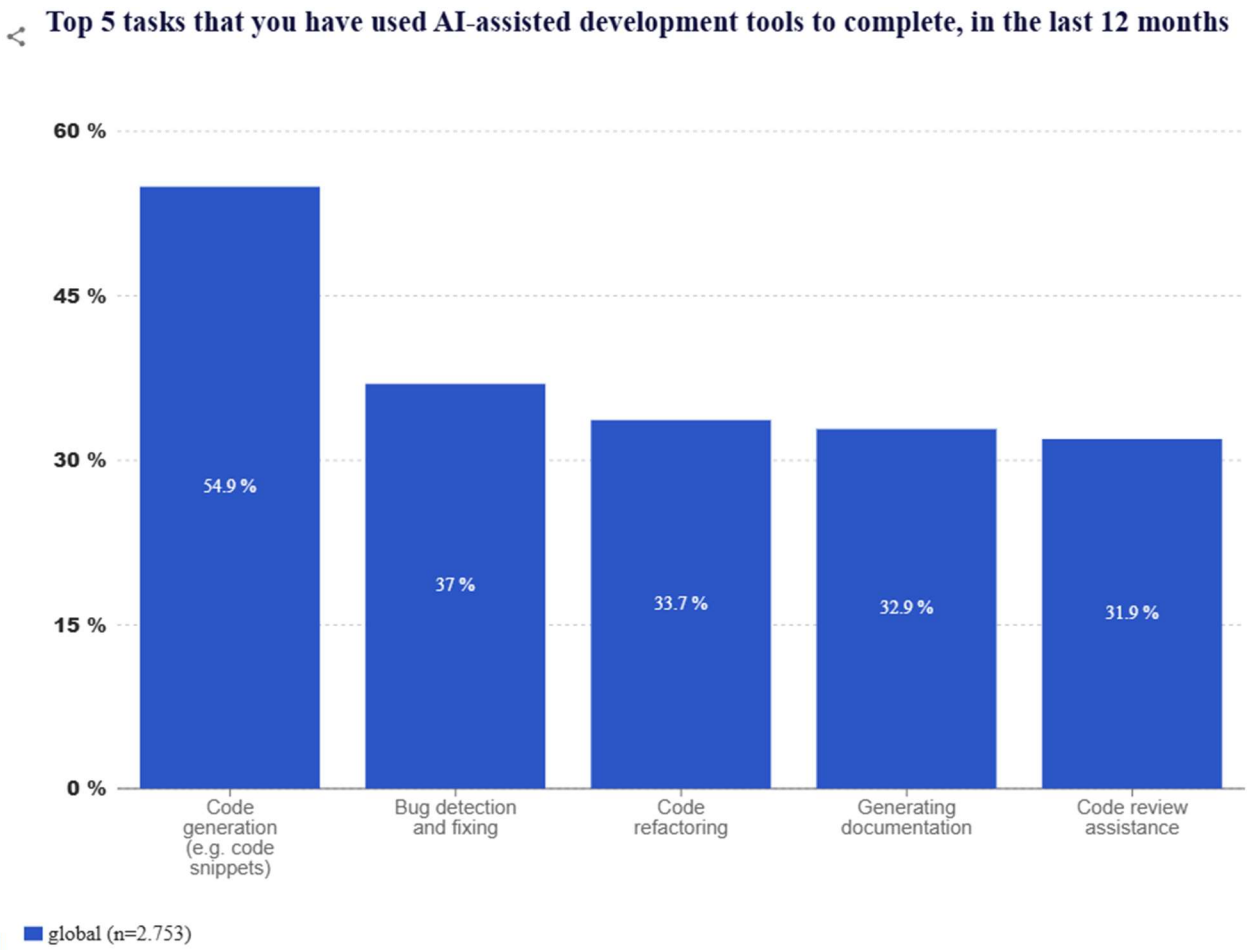
Areas of Software Development, 2024



https://www.developernation.net/developer-reports/dn26/

Developer Nation is a global developer community helping software creators grow throughout their coding journey

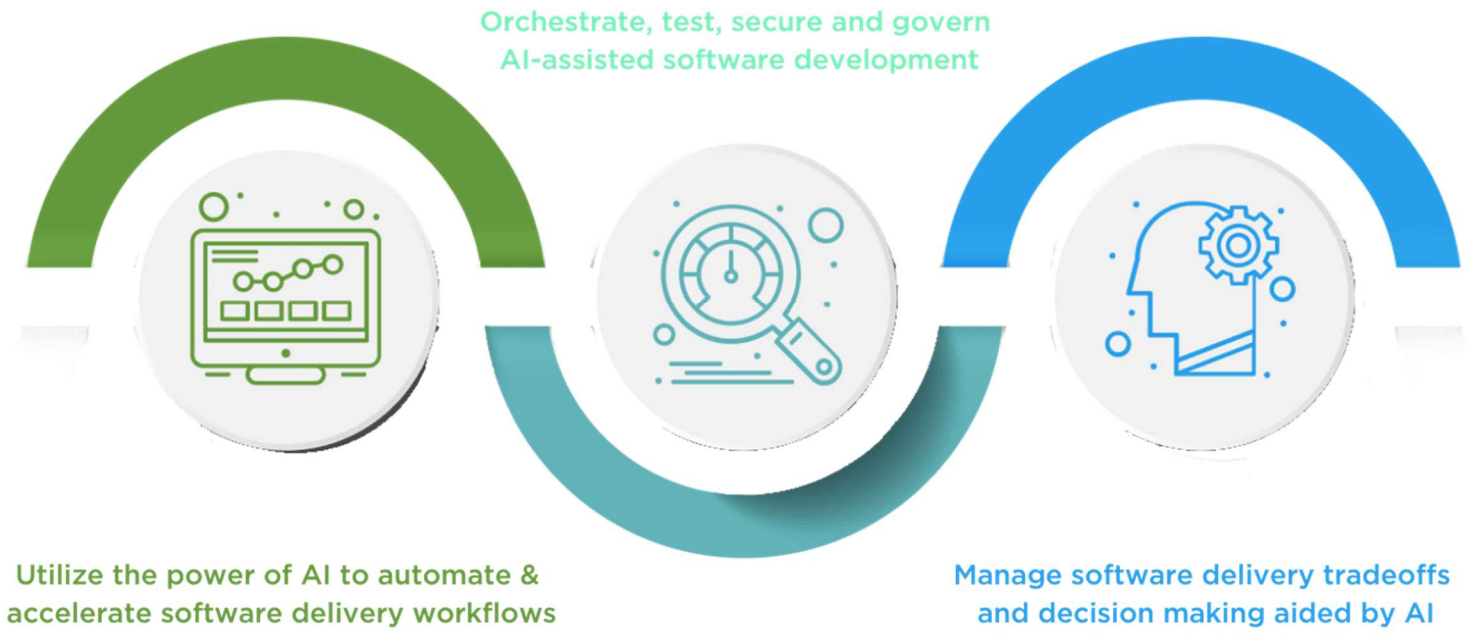
AI-assisted Development Tools to Complete Tasks, 2024



https://www.developernation.net/developer-reports/dn26/

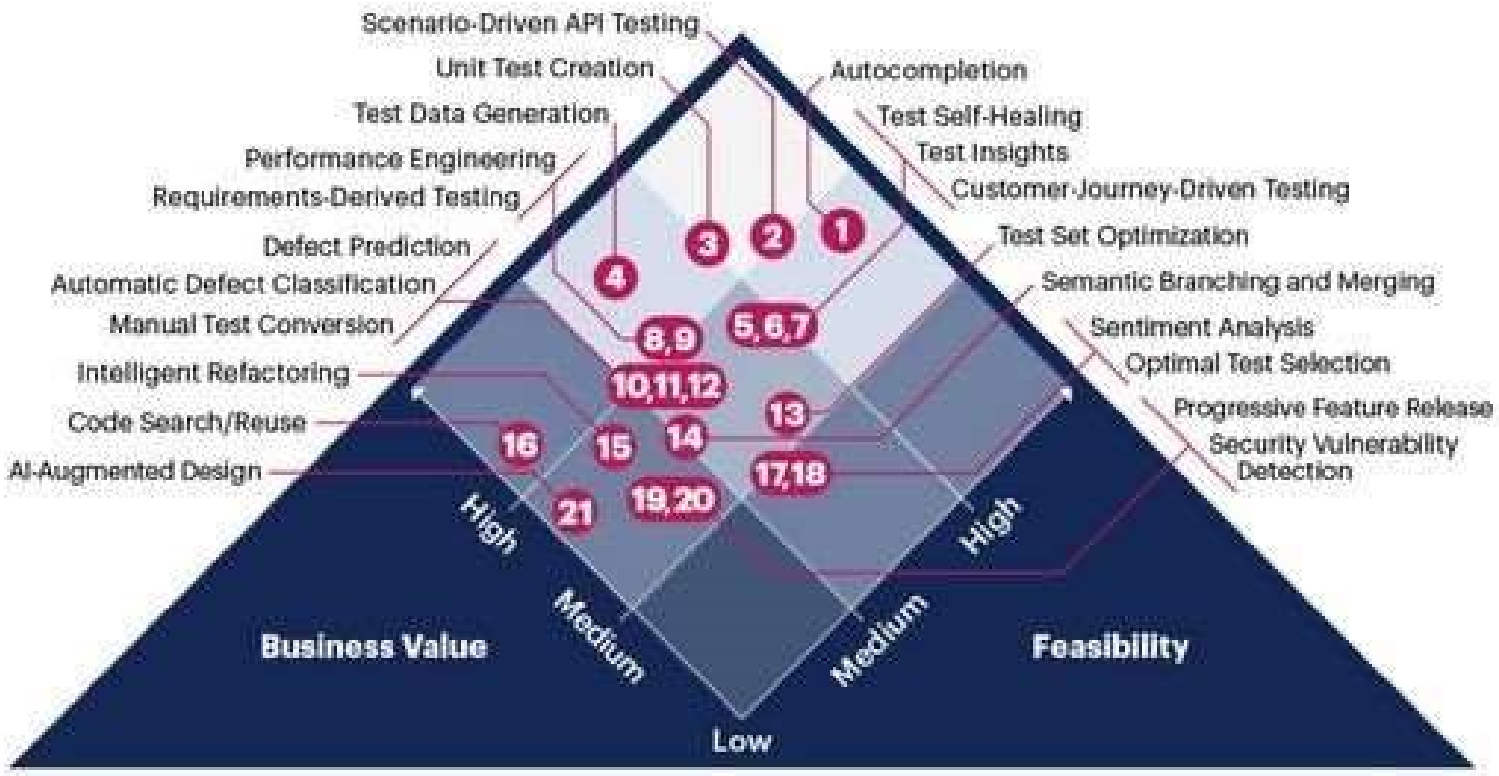
A third replied.

AI-assisted Software Lifecycle, 2024



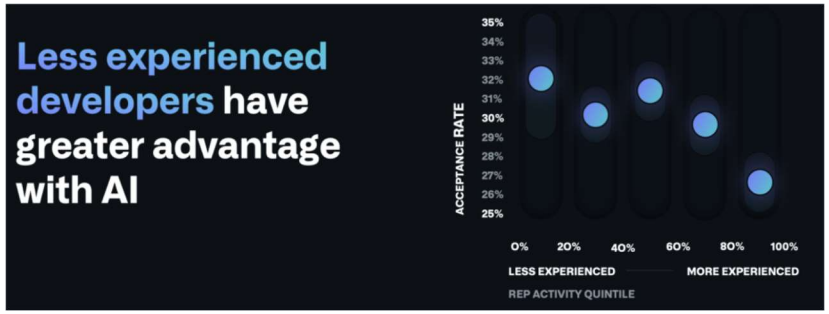
https://digital.ai/solutions/ai/

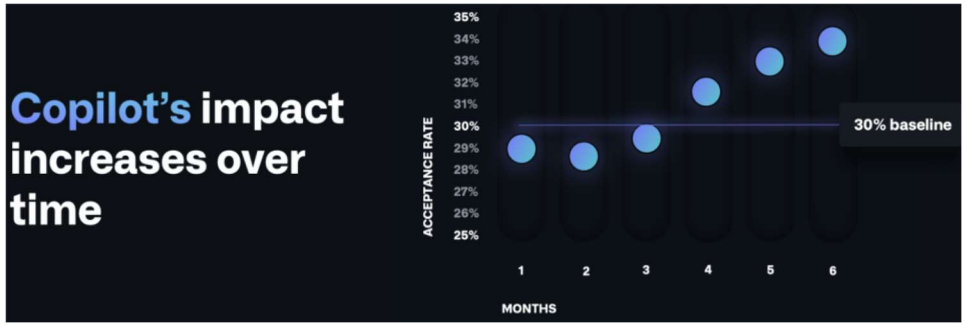
AI Use Cases for Development and Testing, 2020

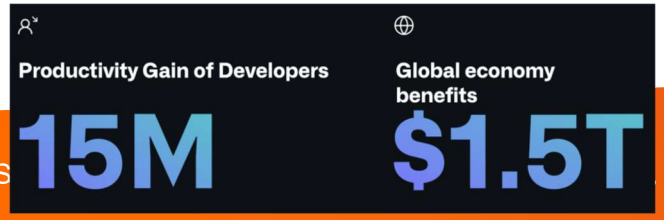


https://www.linkedin.com/pulse/artificial-intelligence-use-cases-development-testing-herschmann/?trackingId=xvKre4wvSiu40dbBlHe4Tw%3D%3

GitHub Productivity Gains of AI in SDLC, 2023







https://github.blog/2023-06-27-the-economic-impact-of-the-ai-powered-developer-lifecycle-and-lessons-from-github-copilot/

- GitHub Copilot is turbocharging developer productivity
- Less experienced developers benefit more
- Faster, happier developers
- Using 30% productivity enhancement and projected number of 45 Mio developers in 2030
  - additional 15 million "effective developers" to worldwide capacity by 2030 could boost global GDP by over $1.5T
  - 100.000 GDP boost per developer

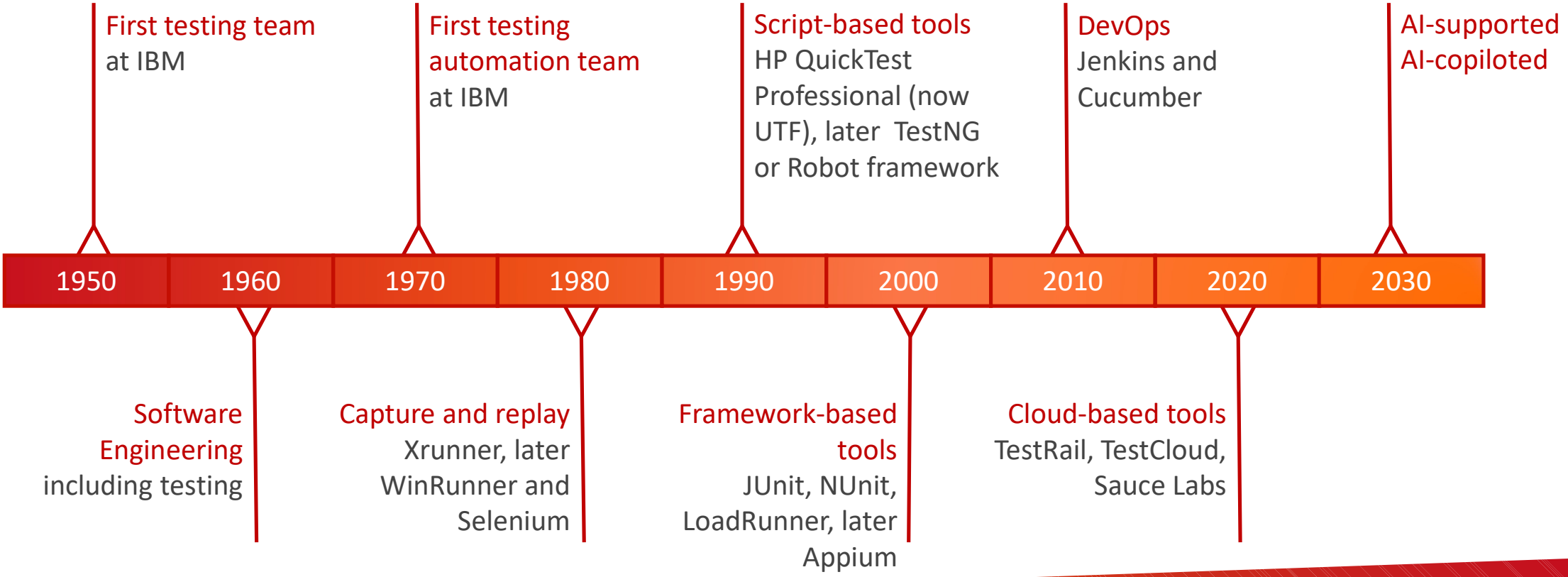
Top Areas for AI in SDLC, 2023

I would delegate it	I am not sure yet	I would still do it myself	
56%	23%	21%	Writing code comments or code documentation
56%	26%	18%	Writing tests
55%	26%	19%	Searching for code fragments inside the codebase
50%	23%	27%	Writing commit messages
46%	23%	31%	Internet searches
35%	34%	31%	Performing actions in CLI
34%	31%	35%	Performing code reviews
34%	31%	35%	Refactoring
31%	28%	41%	Understanding recent code changes
30%	30%	40%	Debugging
23%	26%	51%	Understanding the code
17%	28%	54%	Writing code

https://github.blog/2023-06-27-the-economic-impact-of-the-ai-powered-developer-lifecycle-and-lessons-from-github-copilot/

- Google, 2022: ML-driven assistance led to reducing coding iteration time by 6%, i.e., shrinking “time between builds and tests”
- JetBrains, 2023: most common way for developers to use an AI assistant is to ask general questions about software development using natural language

History and Future of Software Test Automation



Uptake of AI-based methods in Software Testing Tools

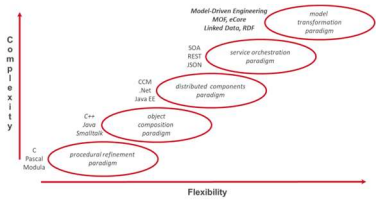
Tool	Key Features	Focus Areas
Testim.io	Machine learning for test authoring and maintenance	Automated UI tests
Applitools	Visual AI for automatic visual validation	Visual regression testing, cross-browser/device testing
Functionize	NLP for test creation, ML for maintenance	End-to-end tests, test analytics
Sauce Labs	AI and ML for visual testing and anomaly detection	Comprehensive testing suite, visual testing
mabl	Machine learning for end-to-end testing automation	Visual regressions, JavaScript errors, link checks
ReTest	AI for difference testing	Regression testing without explicit assertions
Parasoft	AI and ML for noise reduction in static analysis	Static analysis, test optimization
Sealights	AI and analytics for test process insights	Test quality, risk assessment, resource optimization
TestCraft	AI for test maintenance and execution	Codeless Selenium test automation
Katalon Studio	AI for healing test objects and UI test automation enhancement	UI test automation

Selected insights into test case design methods

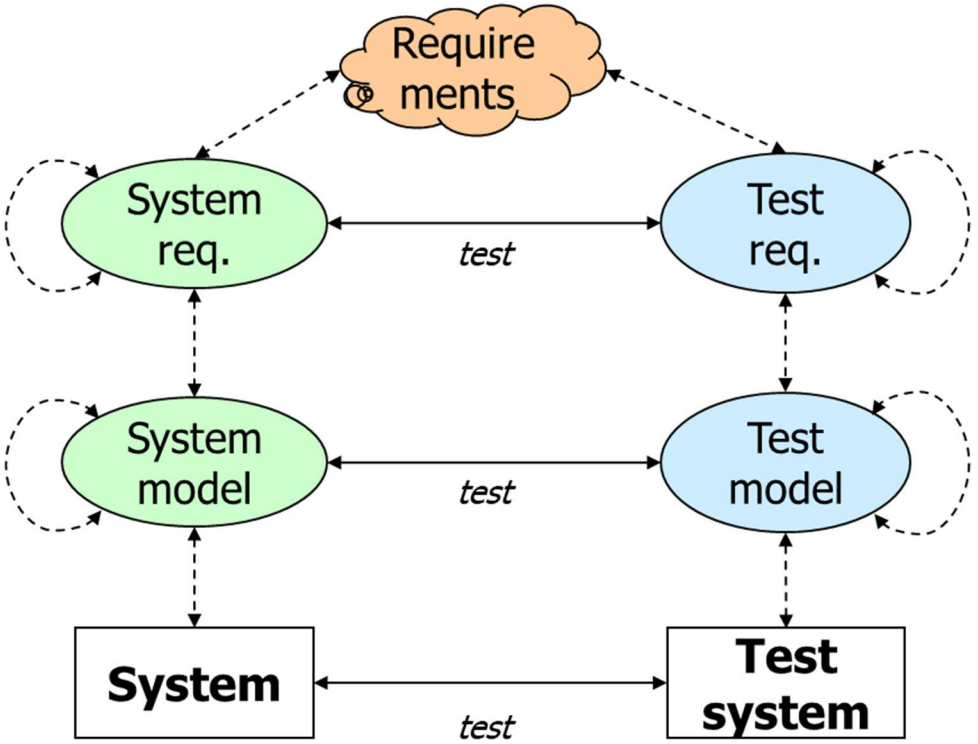
- Xiao *et al.* “Software testing by active learning for commercial games” to generate tests for commercial games
  - learner receives as input samples of input/output pairs extracted from the game engine
  - as output, it yields a model of the game’s expected behavior
- Mariani *et al.* “Automatic testing of GUI-based applications” fro
  - learner receives as input an initial test suite, GUI actio
  - as output, it produces a behavioral model from wh
- hoi *et al.* “Guided GUI testing of android apps with mi nces of test inputs for Android apps
  - learner receives as input sequences of actions extr
  - as output, it gives a model representing the GUI of the app
- Aarts *et al.* “Improving active mealy machine learning for protocol conformance testing”
  - learner receives sequences of input/output pairs
  - as output, it yields a Mealy machine model representing the behavior of the SUT

Models from system executions
are learned
from which
tests are being generated
(MBT v1)

Software Testing before the AI Era

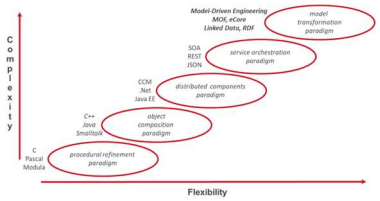


Model-based software engineering

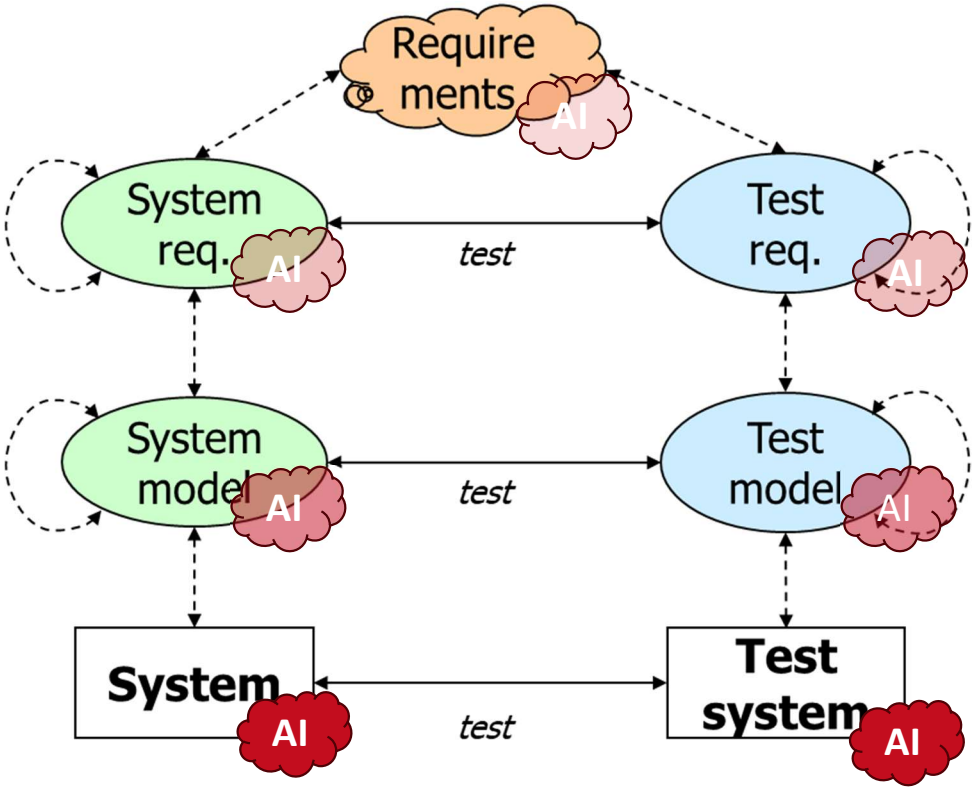


Model-based testing v3

Software Testing in the AI Era ?



Model-based software engineering



Model-based testing v4

Status of AI-based testing research

Anna, Trudova., Michal, Dolezel., Alena, Buchalcevova. (2020). Artificial Intelligence in Software Test Automation: A Systematic Literature Review. 181-192. doi: 10.5220/0009417801810192

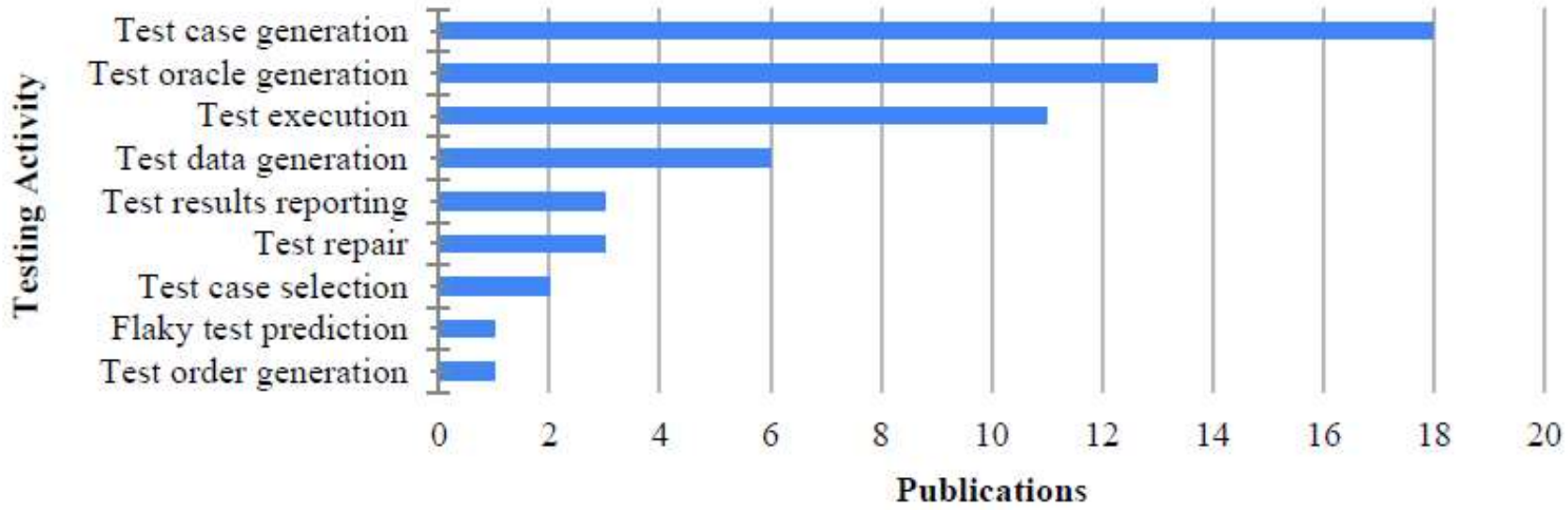


Figure 2: Number of publications by testing activity.

Status of AI-based testing research

Ricca, Filippo, Alessandro Marchetto, and Andrea Stocco. "AI-based test automation: A grey literature analysis." *2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 2021.

TABLE I: Taxonomy of main TA problems by practitioners.

PROBLEM	#
Test Planning (22)	
Critical paths identification	13
Planning what to test	7
Planning long release cycles	2
Test Design (8)	
Programming skills required	5
Domain knowledge required	3
Test Authoring (109)	
Manual code development	52
Manual API test development	7
Manual data creation	19
Test object identification	13
Cross-platform testing	10
Costly exploratory testing	5
Locators for highly dynamic elements	1
Test code modularity	1
Accessibility testing	1
Test Execution (71)	
Untested code	29
Flakiness	18
Slow execution time	14
Useless test re-execution	4
Scalability	2
Parallelization	2
Low user responsiveness	1
Platform independence	1
Test Closure (47)	
Manual debugging overhead	18
Costly result inspection	10
Visual analysis	19
Test Maintenance (82)	
Manual test code migration	3
Bug prediction	11
Fragile test script	10
Regression faults	2
Costly visual GUI regression	8

TABLE II: Taxonomy of AI Solutions to TA problems.

SOLUTION	#
Test Generation (125)	
Automated test generation	29
Automated test generation using machine translation	11
Automated test generation from user behaviour	11
Automated test generation from API calls	6
Automated test generation from mockups	3
Automated test generation using crawling	2
Automated data generation	22
Robust element localization	13
Dynamic properties recognition based on user behaviour	8
Automated exploratory testing	7
Object recognition engine	6
Mock generation	3
Self-learning	2
Automated API generation	1
Page object recognition	1
Oracle (38)	
Visual testing	38
Debugging (62)	
Intelligent test analytics	17
Automated coverage report	14
Noticeable code changes identification	12
Runtime monitoring	10
Flaky test identification	7
Bad smell identification	1
Decoupling test framework from host	1
Maintenance (81)	
Self-healing mechanisms	43
Self-healing test scripts	24
Smart locators	19
Intelligent fault prediction	12
Test selection intelligent test case re-execution	12
Intelligent waiting sync	5
Intelligent test case prioritization	4
Automated identification environment configurations	3
Pattern recognition	1
Remove unnecessary test cases	1
Unspecified	91

Status of AI-based testing research

N. Alshahwan, M. Harman and A. Marginean, "Software Testing Research Challenges: An Industrial Perspective" *2023 IEEE Conference on Software Testing, Verification and Validation (ICST)*, Dublin, Ireland, 2023, pp. 1-10, doi: 10.1109/ICST57152.2023.00008.

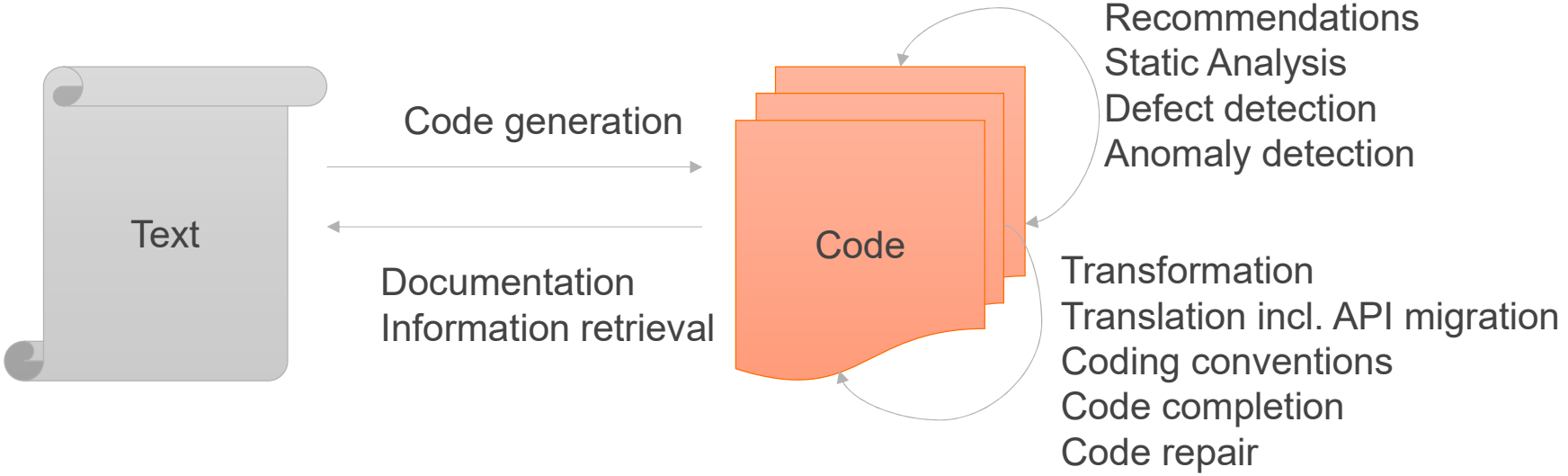
TL;DR

- industrial perspective on challenges in software testing and optimisation (improvement) research:
  - Software Test Generation
  - Automated Repair and Improvement
  - Automated Transplantation and Refactoring
- witnessed an increasing intensity of work on AI-based optimisation techniques for improving software testing

Insights

- automated test data generation is a natural complement to generative AI
  - but query engineering problem as a new research target
- reference to Sobania, Dominik, et al. "An analysis of the automatic bug fixing performance of chatgpt." 2023 IEEE/ACM International Workshop on Automated Program Repair (APR). IEEE, 2023.
  - ChatGPT outperforms other bug fixing approaches (analysed with QuixBugs benchmark)

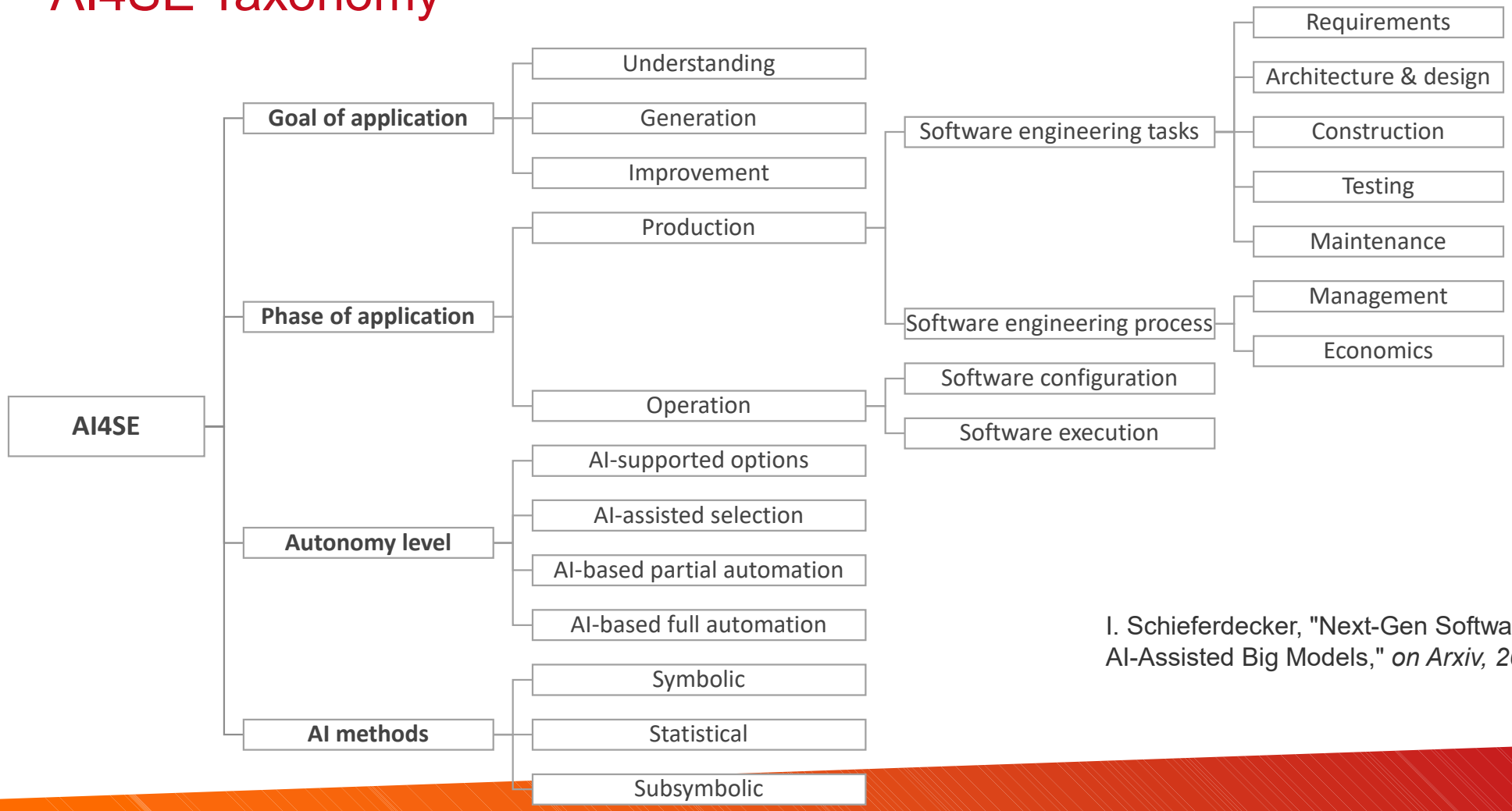
More General: AI in Software Engineering



Recent Results for AI-Support in Software Engineering

Tasks	Understanding	Generation	Improvement
Requirements	(Körner et al, 2014)	(Körner et al, 2014)	(Körner et al, 2014) (Perini et al, 2012)
Architecture & Design	(Bhat et al, 2019)	(Bhat et al, 2019)	-
Code	(Ceran et al, 2023) (Guo et al, 2020) (Gupta, Sundaresan, 2018)	(Bird et al, 2023) (Svyatkovskiy et al, 2020)	(Drain et al, 2021) (Guo et al, 2020) (Bader et al, 2019)
Construction	(Lenz et al, 2013)	(Tufano et al, 2022)	(Zhao et al, 2015) (Lenz et al, 2013)
Configuration	(Tatineni et al, 2021)	(Tatineni et al, 2021)	(Tatineni et al, 2021)
SE Process	(Lin et al, 2021)	-	(Spieker et al, 2017)

AI4SE Taxonomy



I. Schieferdecker, "Next-Gen Software Engineering: AI-Assisted Big Models," *on Arxiv, 2024*

Big Code as Basis for AI in SE

Allamanis, Miltiadis, et al. "A survey of machine learning for big code and naturalness." *ACM Computing Surveys (CSUR)* 51.4 (2018): 1-37.

→ **The naturalness hypothesis.** Software is a form of human communication; software corpora have similar statistical properties to natural language corpora; and these properties can be exploited to build better software engineering tools.

Pudari, Rohith, and Neil A. Ernst. "From copilot to pilot: Towards AI supported software development." *arXiv preprint arXiv:2303.04142* (2023).

→ **Necessity for elevated abstraction levels** to enhance the efficacy of AI-assisted automated coding

Hamilton, William L., Rex Ying, and Jure Leskovec. "Representation learning on graphs: Methods and applications." *arXiv preprint arXiv:1709.05584* (2017).

→ **Formal graph structures of SE models** can be leveraged to facilitate the preparation of models for AI/ML applications

Big Models as Basis for AI in SE

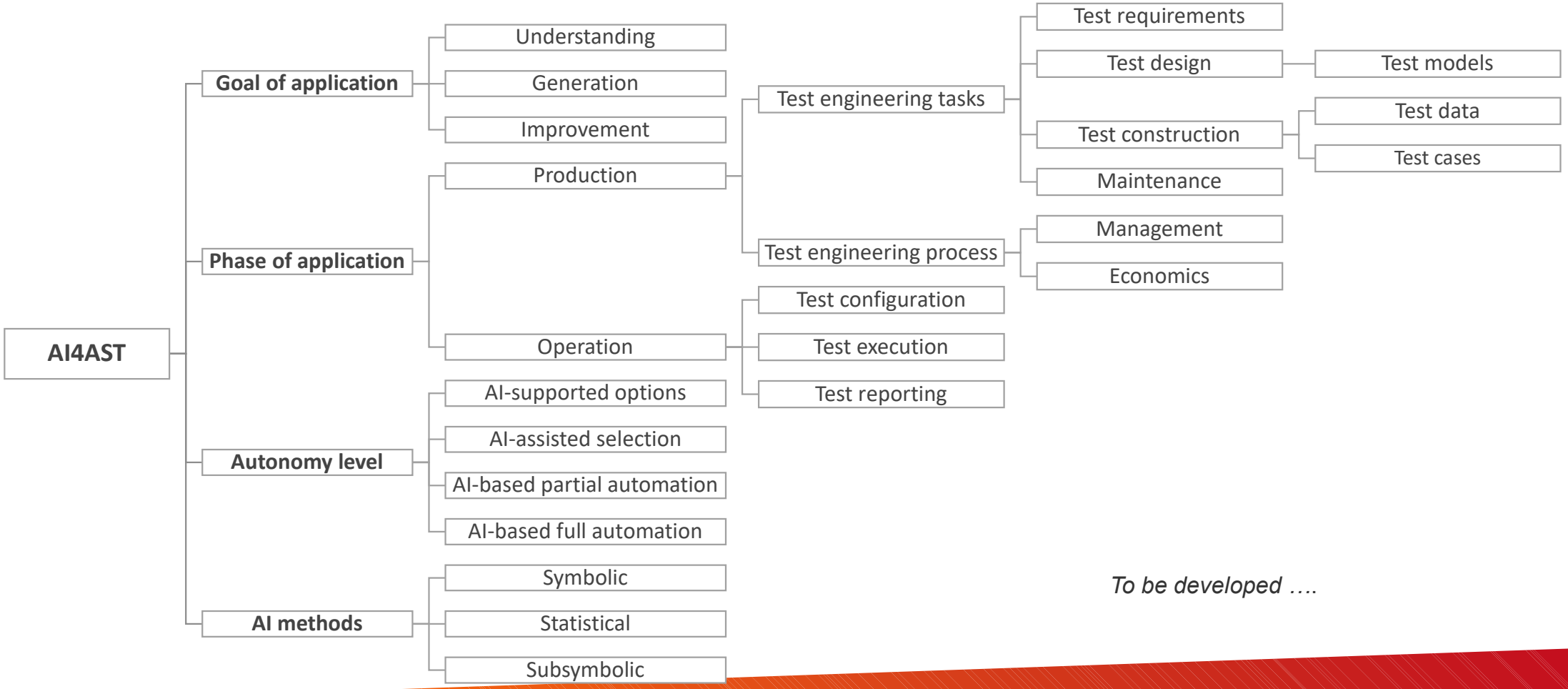
→ **The modelling hypothesis.** Software is (also) a formalized communication between humans and computers; model corpora, software corpora, and natural language corpora have similar statistical properties; the properties of model corpora can be employed to develop more efficacious software engineering tools

→ **Opportunities for MDSE and AI in SE as well as for AI in Software Test Automation**

1. The provision of large amounts of top-down models on coding platforms
2. The generation of bottom-up models from big code
3. The formation of Big Models as a basis for more advanced empirical MDSE and further improvements of MDSE
4. The application of AI methods on Big Models and for Big Models
5. Pair modelling in MDSE

→ **Next Generation Software Engineering Using Big Models and AI**

Back to Testing: AI4(A)ST Taxonomy



To be developed ….

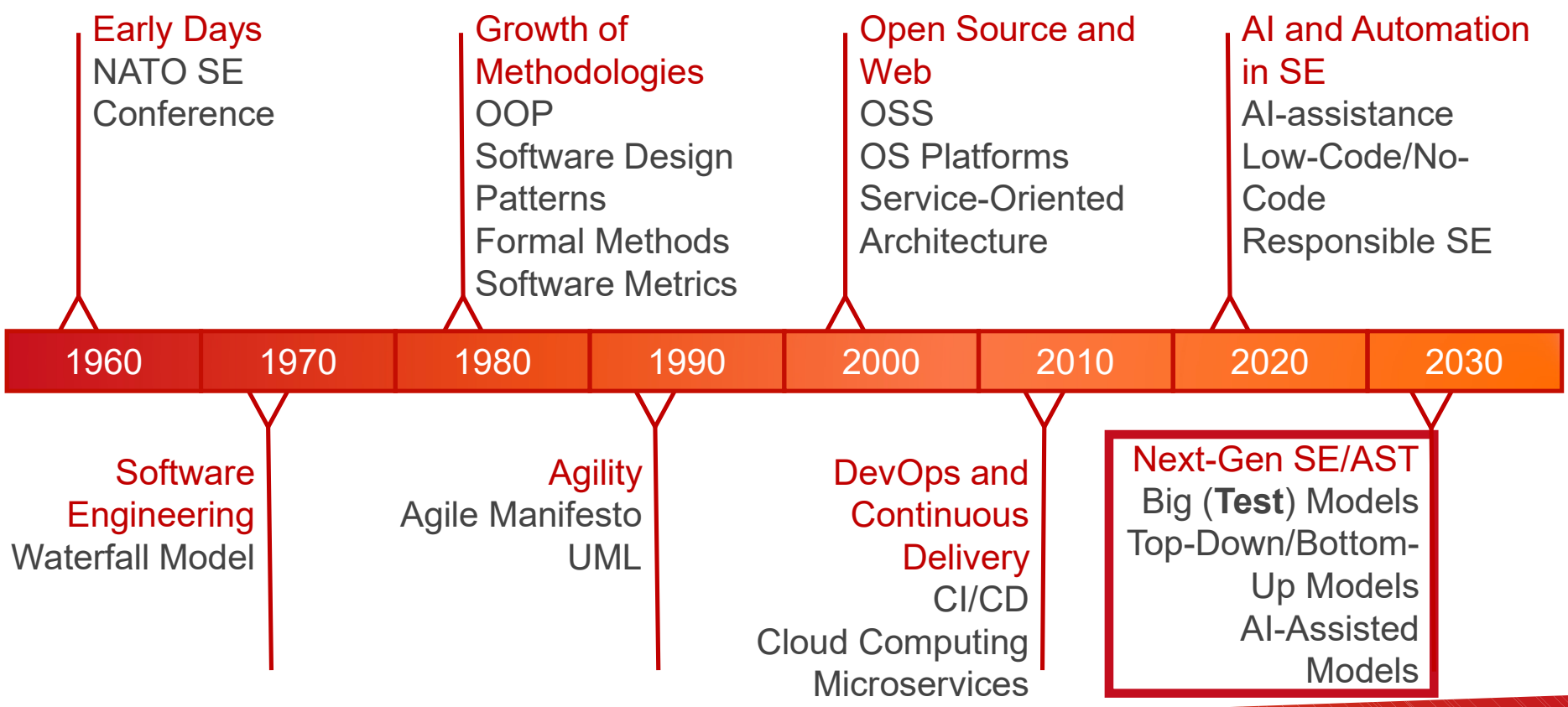
AI-Support for MBT

Tasks	Understanding	Generation	Improvement
MBT v1	(Job, 2021)	(Khan et al, 2024) (Naimi et al, 2024)	(Naimi et al, 2024)
MBT v2			
MBT v3			

Summary

- Since long time
  - Mutation-based testing
  - Fuzz testing
  - Metamorphic testing
  - Evolutionary testing
  - …
- Currently
  - ML-based testing, transfer learning for test generation – „resembles“ exploratory testing
  - NLP for test generation – „resembles“ scenario-based testing
  - Design and coding support – „resembles“ pair-programming
- Soon: AI for test prioritization, test effort estimation, test process optimization
- Later: AI for MBT

History and Future of Software Engineering **and Testing**



To the next 50 TAV meetings!

Thank you for your attention.

If you want to follow up:

Automated Software Testing with AI paper collection:
https://www.researchrabbitapp.com/collection/public/467QMKM8ZX

I used beside other the following tools :