

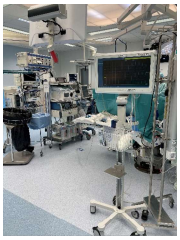
1

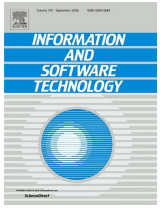
Who am I?

Programmer, Architect, Author, Professor

- Software Engineering at Computer Science and Engineering
  - Integrated department from Chalmers and University of Gothenburg
- Specialization in
  - Machine learning in software engineering
  - Federated learning in SE and ICU
  - Autonomous artificial intelligence-based measurement, machine reasoning
- Editor-in-Chief of Information and Software Technology
  - AI engineering
  - Empirical software engineering
  - Software Technology
- Main collaborations
  - Ericsson, Volvo Cars, Scania
  - Software center
  - Sahlgrenska University Hospital (NICU)
  - AI Sweden

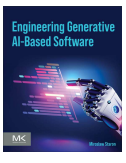












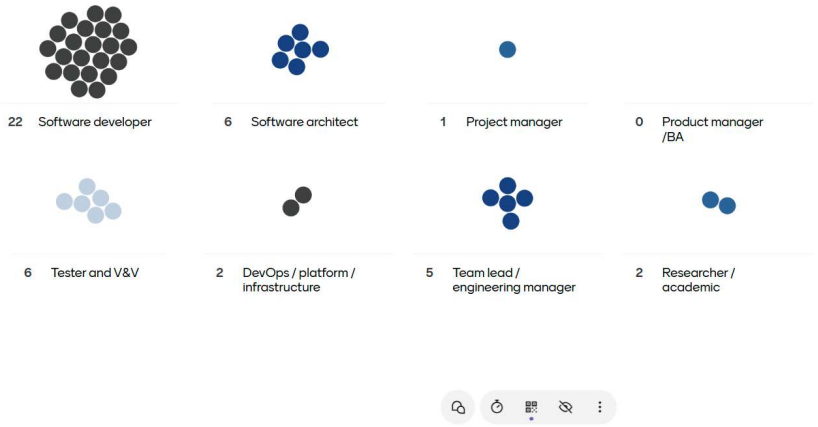


Software Center

www.software-center.se

2

Who are you?





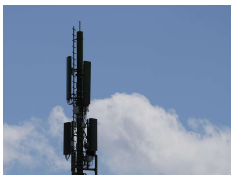
Software Center

www.software-center.se

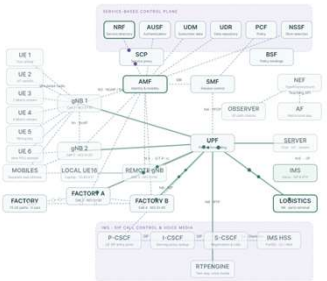
3

A Future Software Engineer is …

- … a great software developer
  - knows the technology
- … a domain expert
  - knows the “why”
- … a product manager
  - knows the “what”
- … an infrastructure specialist
  - knows the “how”





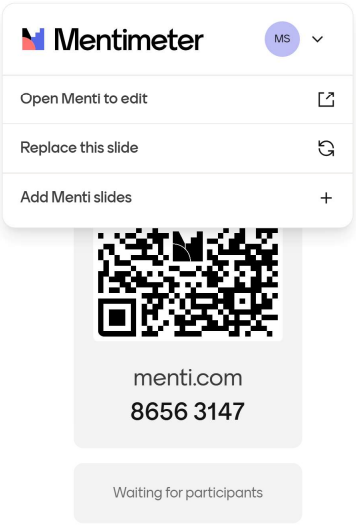


Software Center

4

What makes a great software developer today?

0	Writes clean, maintainable code
0	Deep knowledge of languages, frameworks and tools
0	Understands the problem domain and the users
0	Designs good architectures and abstractions
0	Tests and verifies thoroughly
0	Communicates and collaborates well
0	Uses AI tools effectively



5

Software 1.0

Source code

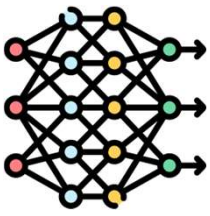
Computer (compiler)



Software 2.0

Weights

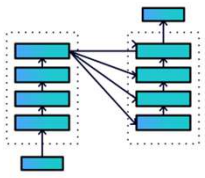
Neural network



Software 3.0

Prompts

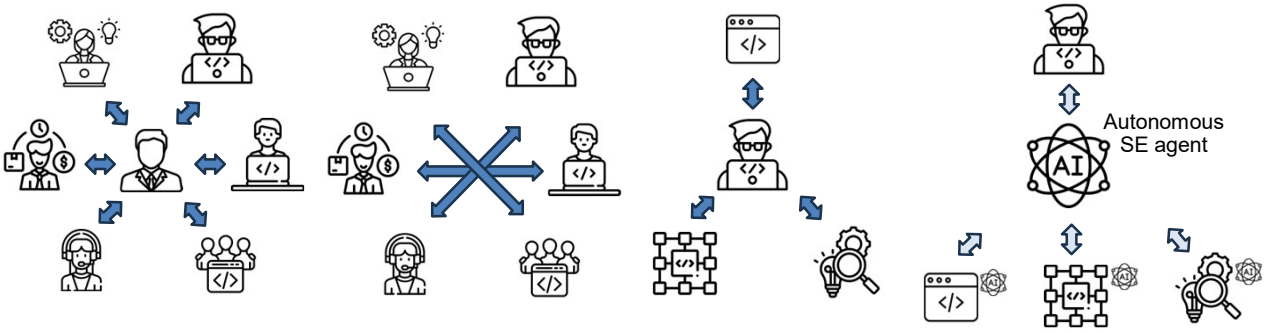
LLM, generative AI



Based on the slides from (2) Andrej Karpathy: Software Is Changing (Again) - YouTube

6

Four types of software eng. teams



Type I: Specialists

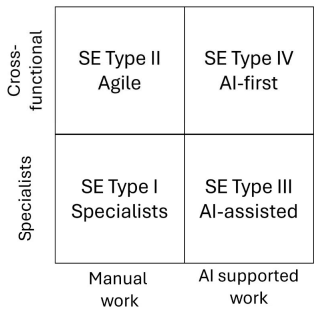
Type II: Agilists

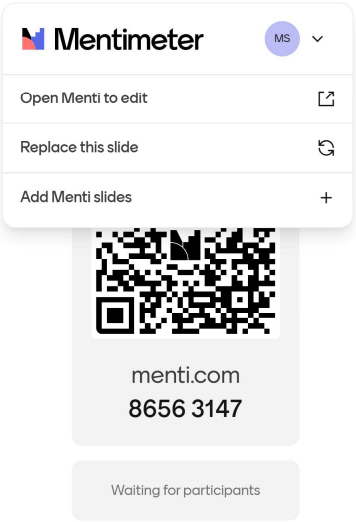
Type III: AI-Assisted

Type IV: AI-first

7

Drop a pin where you think your team is



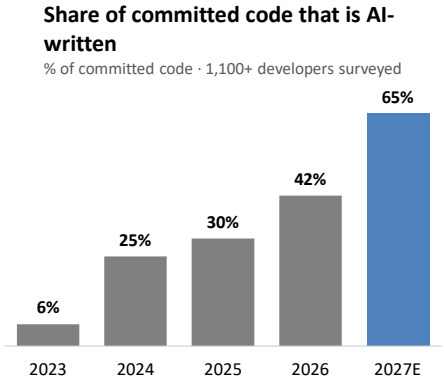


8

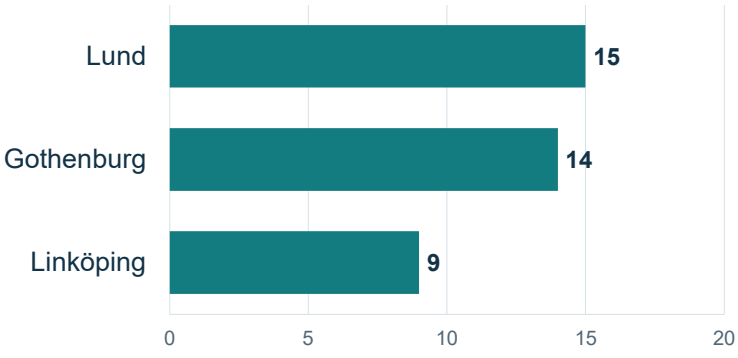
AI writes 42% of committed code today

- Sonar's 2026 State of Code survey (1,100+ developers): AI accounts for 42% of all committed code
- Developers expect that share to reach 65% by 2027 — up from 6% in 2023
- 72% of developers who have tried AI coding tools now use them daily
- But 96% do not fully trust AI-generated code, and only 48% always verify it

Source: Sonar, 2026 State of Code Developer Survey (Jan 2026). 2027 is a developer projection; 2024 (Google), 2025 (Microsoft)



Three industrial workshops



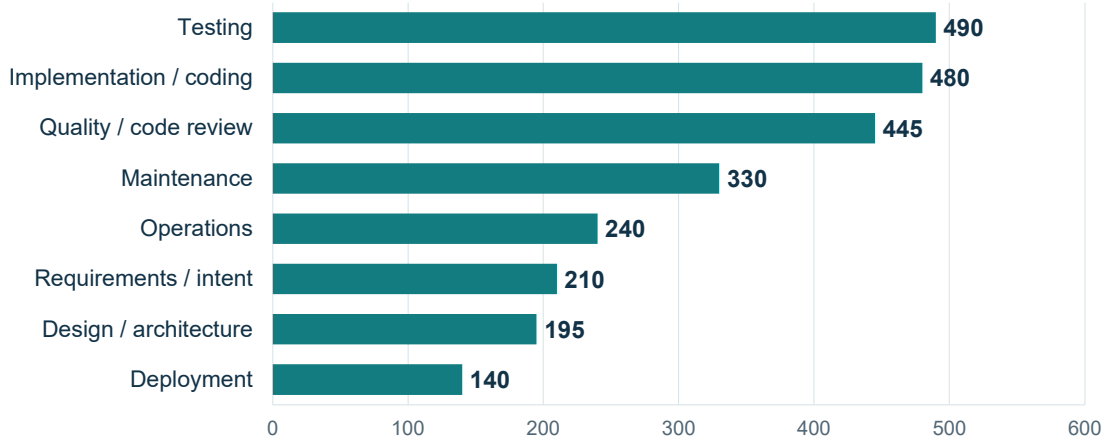
38

survey respondents in the merged analysis

Three industrial workshops

One preceding academic workshop

Where will AutoEvolve happen first



11

#2

Domain expert

12

How well do you understand the domain you work in?

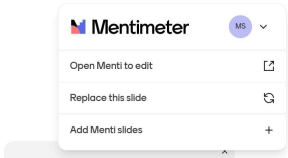
I could basically do my user's job

I understand the standards and regulations of my field

I follow the literature of the domain (e.g., cars/trucks in automotive, mobile in telecom)

I know exactly where in the domain my product is situated and why

Strongly disagree — Strongly agree



menti.com
8656 3147

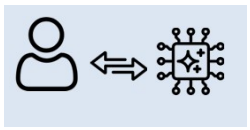
Waiting for participants

Software Center

www.software-center.se

13

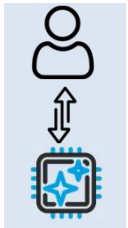
Evolution becomes an operational capability of the software itself





Vibe-coding or Software-on-demand:

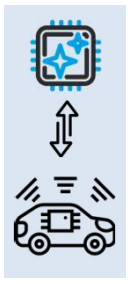
- Humans develop software with the help of agents
- Development external to the software





Agentic software:

- Agents develop software
- Agents are external to the software

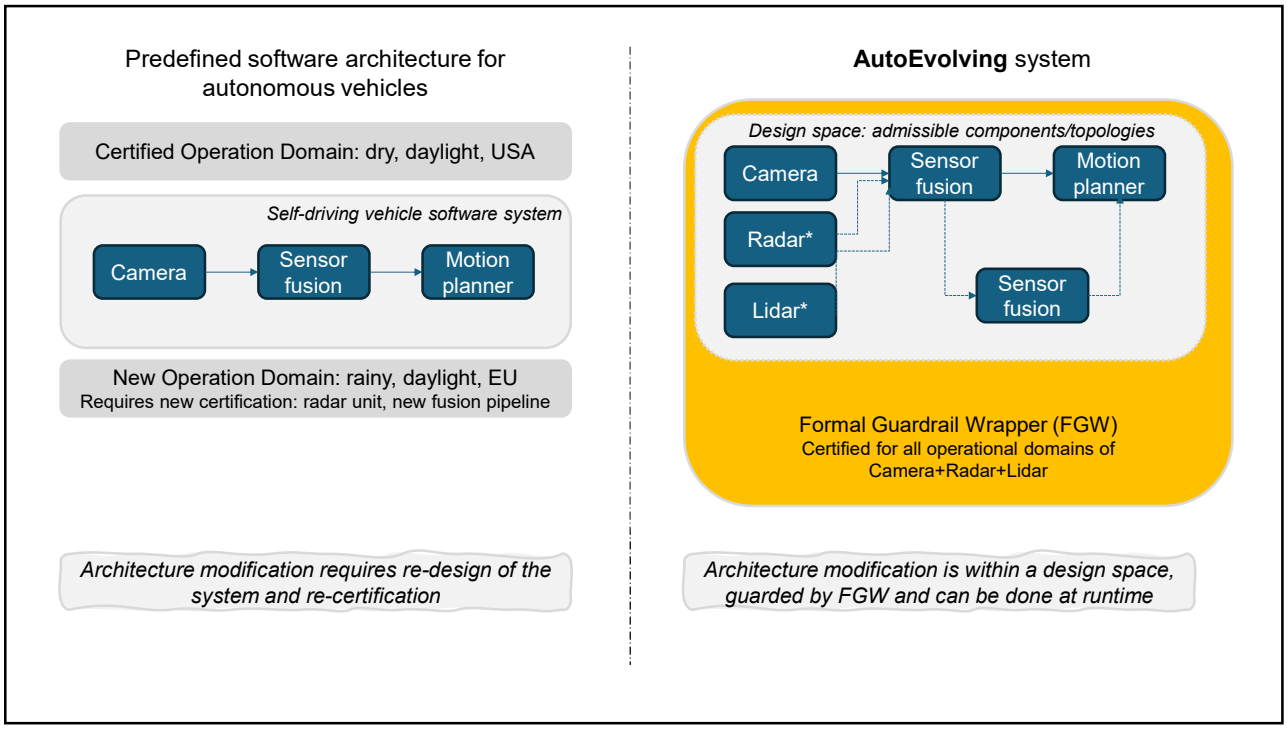


Autoevolving software:

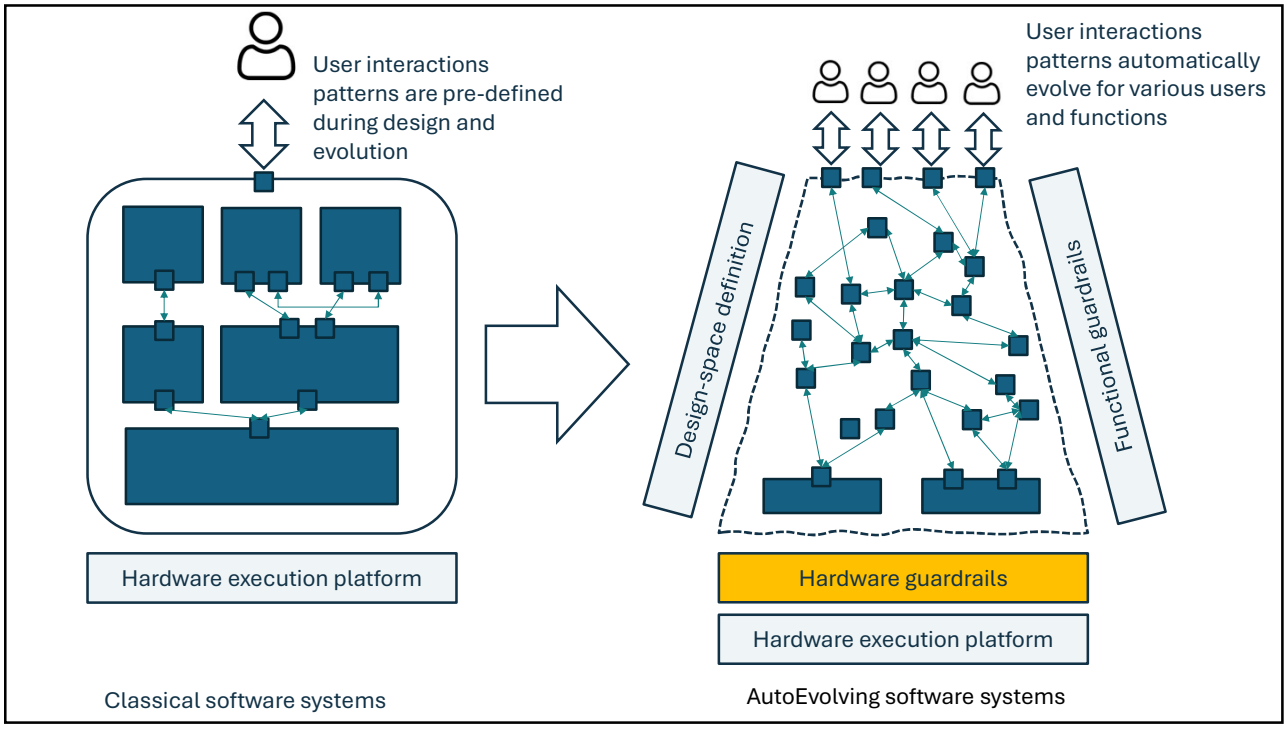
- Agents develop software
- Agents are internal to the software

Generations of software

14


Predefined software architecture for autonomous vehicles
Certified Operation Domain: dry, daylight, USA
Self-driving vehicle software system
Camera
Sensor fusion
Motion planner
New Operation Domain: rainy, daylight, EU
Requires new certification: radar unit, new fusion pipeline
Architecture modification requires re-design of the system and re-certification
AutoEvolving system
Design space: admissible components/topologies
Camera
Sensor fusion
Motion planner
Radar*
Lidar*
Sensor fusion
Formal Guardrail Wrapper (FGW)
Certified for all operational domains of Camera+Radar+Lidar
Architecture modification is within a design space, guarded by FGW and can be done at runtime

15


User interactions patterns are pre-defined during design and evolution
Hardware execution platform
Classical software systems
User interactions patterns automatically evolve for various users and functions
Design-space definition
Functional guardrails
Hardware guardrails
Hardware execution platform
AutoEvolving software systems

16

Design-level intelligence

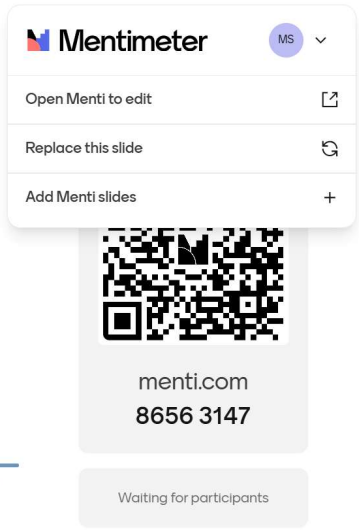
code-level intelligence is the ability of an AI system to **write, understand, and repair source code** so that it compiles, passes its tests, and does what the developer intended

design-level intelligence is the ability of an AI system to **reason about, modify, and repair software designs** while preserving their intended behavioral properties

#3

Product manager

How often do you know *why* a feature is being built?



0	0	0	0
Always, I know the user need and expected impact	Usually, I get the gist	Sometimes	Rarely, I get a ticket and build it

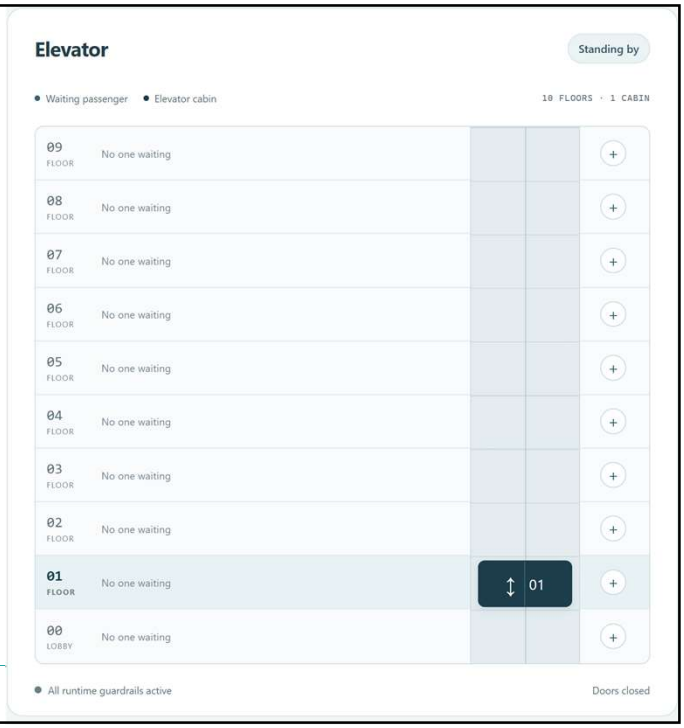
Software Center

www.software-center.se

19

Elevator example

- Elevator that moves between different floors with **guardrails**:
  - cannot stop between floors or drop to 0 from any other floor
  - can be stopped by a human, but then needs to be checked for safety
- Safety-critical simulation
- Guardrails for hardware access
- Explicit evolution: from simple C controller to C controller with web interface

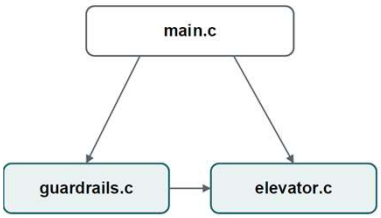


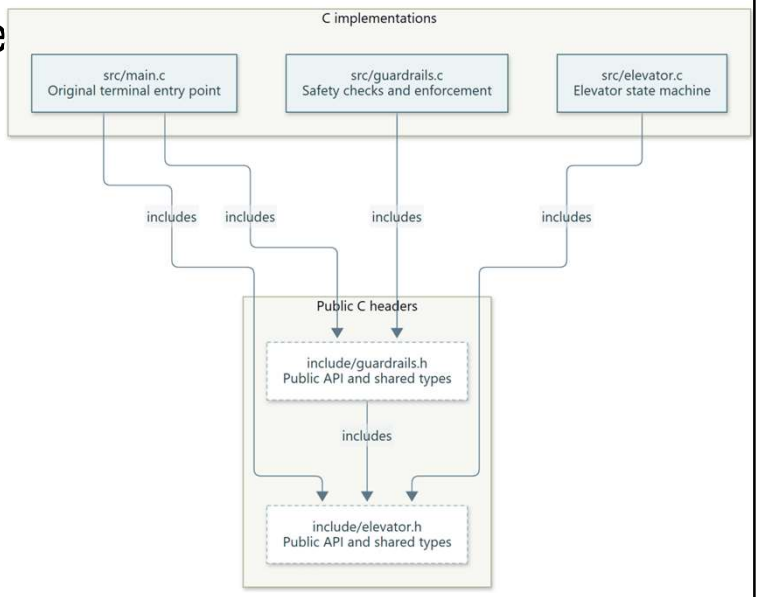
Software Center

20

Initial architecture

- Three components
  - Guardrails
  - State machine
  - Execution/simulation

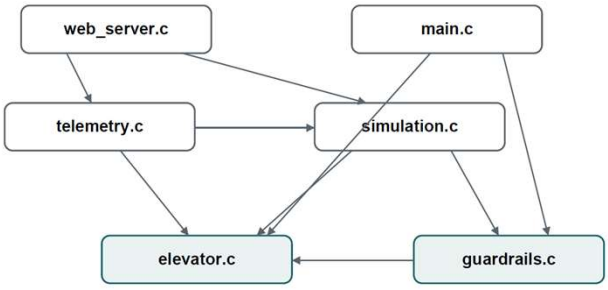


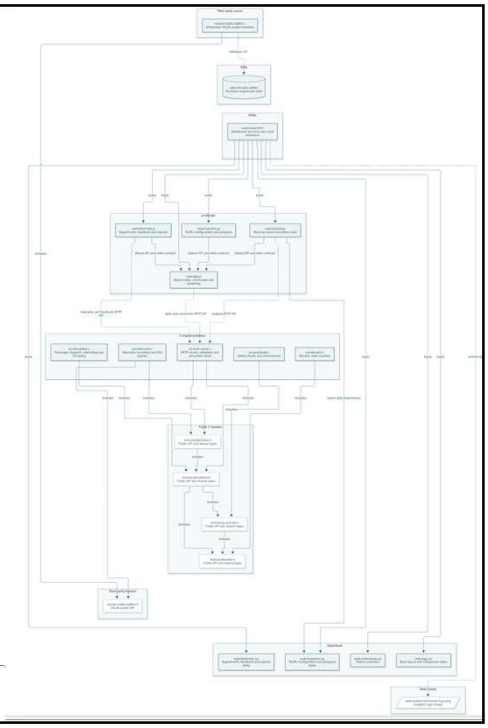


21

Evolved system, telemetry

- Guardrails and the original simulation was preserved
- Additions are done to non-safety-critical code

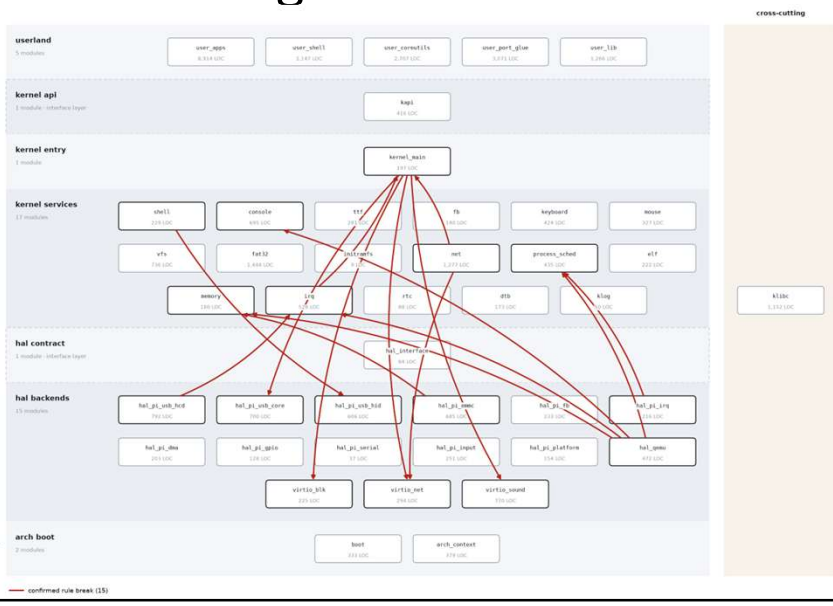




22

VibeOS: AI agents do design

- AI agents do not respect the design
- AI agents treat all components as equal



Software Center

23

Adaptation and AutoEvolution

Dimension	Adaptation in a fixed design space (MAPE-K)	AutoEvolution
Available changes	Choose among predefined options	Generate new implementations
Architecture	Preserve the predefined structure	May revise structure and interfaces
Feedback	Select or tune known behaviour	Guide design and implementation
Control	Rules and limits set in advance	Explicit intent and evolving constraints

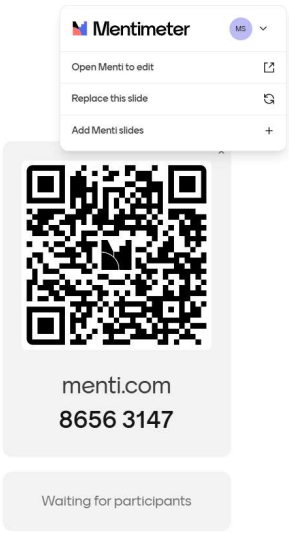
#4

Infrastructure provider

25

Where does the code you write end up running?

- I don't really know, someone else handles it
- In the cloud, set up by an ops or platform team
- In the cloud, set up by me or my team
- On embedded devices or products
- On our own servers
- It depends on the project



26

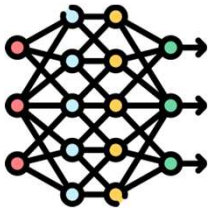
The runtime is the development environment

Software 1.0



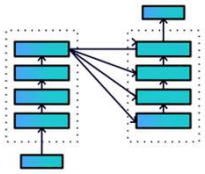
- Deployment and HW
- Program analysis
- Software metrics
- Sw. Designers and Programmers

Software 2.0



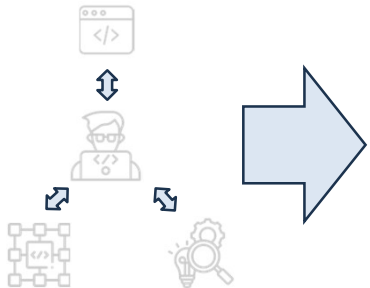
- Infrastructure & Cloud
- Accuracy metrics
- Performance metrics
- Data Scientists and Programmers

Software 3.0

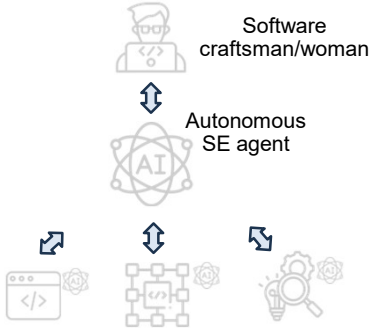


- Infrastructure as code
- Benchmarks
- Probability metrics
- Domain specialists and Programmers

Multi-agentic SE 4.0



AI-assisted SE 3.0

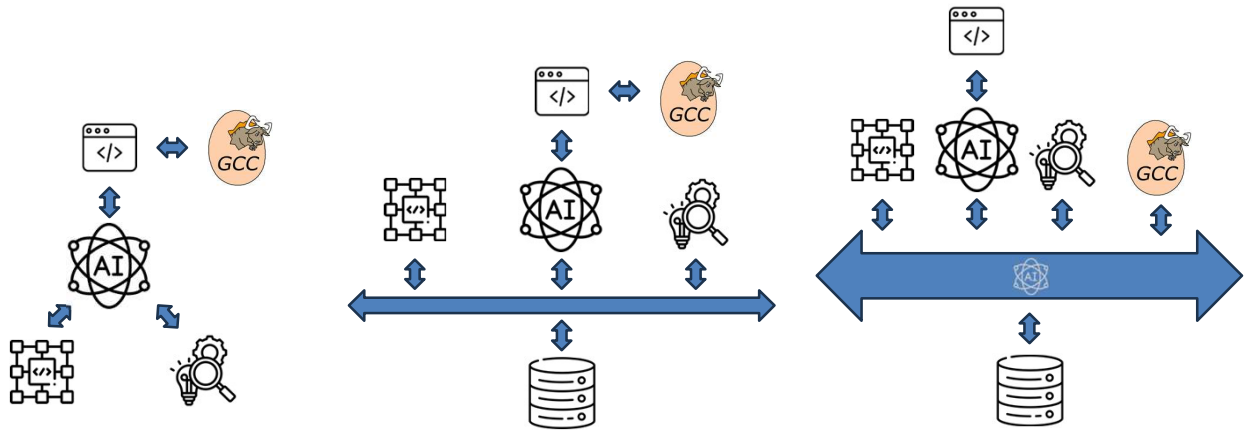


Autonomous SE




```
# AgenticAI Codebase Guide

## Project Overview
This is a research project that implements multi-agent
and analysis. The system creates designer-programmer c
to study iterative code improvement, compilation succe

## Development Setup

## Core Architecture

### Agent System (`agent.py`, `agentC.py`, `agentCompiler.py`)
- **`AgentAI`**: Base class managing Ollama API conversations wi
- **`AgentAIC`**: Extends base with C code compilation and error
communicates with AgentCompiler for validation.
- **`AgentCompiler`**: Specialized for compilation error detecti
gcc to compile extracted C code snippets.
- All agents maintain conversation state via `self.messages` lis
- **`AgentStaticAnalyzer`**: Analyzes C code for static issues u
- **`AgentHuman`**: Facilitates human-in-the-loop conversations
essentially a wrapper for human input.
```

Agent architectures



Star Message bus Middleware

A Future Software Engineer is …

- … a great software developer
  - knows the technology
- … a domain expert
  - knows the “why”
- … a product manager
  - knows the “what”
- … an infrastructure specialist
  - knows the “how”

