Modeling for Agentic AI - Opportunities for the MDE Community in the AgentWare Era

MDE Intelligence Workshop 2025

Davide Di Ruscio

Università degli studi dell'Aquila / Italy

davide.diruscio@univaq.it

SWEN research group

MOSAICO

Main contents of this talk

- The evolution of software development paradigms
- Opportunities and challenges of applying generative AI in software engineering
- The MOSAICO EU project
- Opportunities for the MDE community

Roadmap

The evolution of software development paradigms

Large Language Models for SE

LLM-based Multi-Agent Systems

MOSAICO

The MOSAICO project

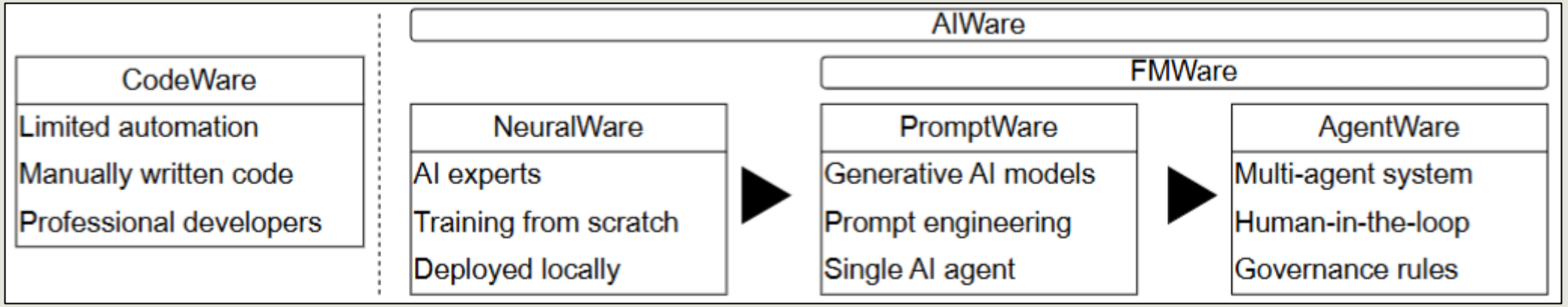
https://mosaico-project.eu/

MDE Opportunities

Conclusion

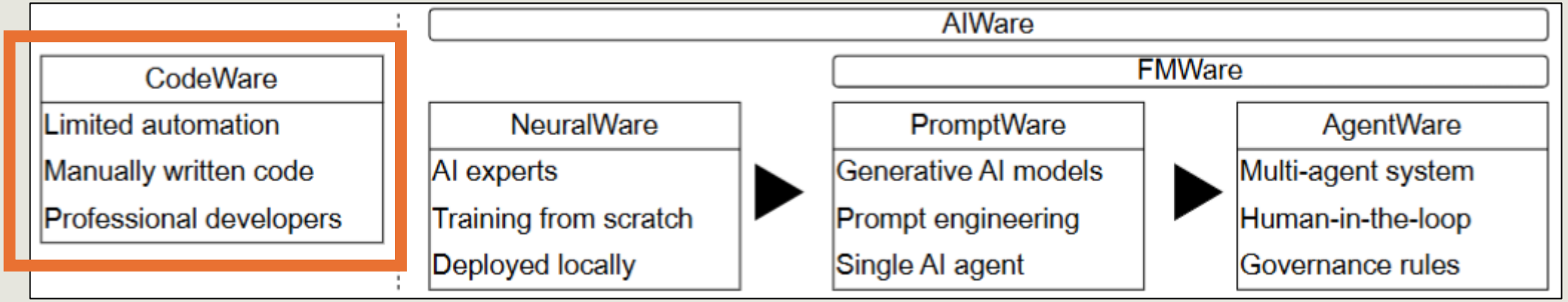
The evolution of software development paradigms

The evolution of software development paradigms



Filipe Roseiro Cogo, Gopi Krishnan Rajbahadur, Dayi Lin, and Ahmed E. Hassan, "**A Tutorial on Software Engineering for FMware**", in: Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering, *FSE 2024*, Porto de Galinhas, Brazil: Association for Computing Machinery, 2024, pp. 710–712, ISBN: 9798400706585, DOI: 10.1145/3663529.3663820.

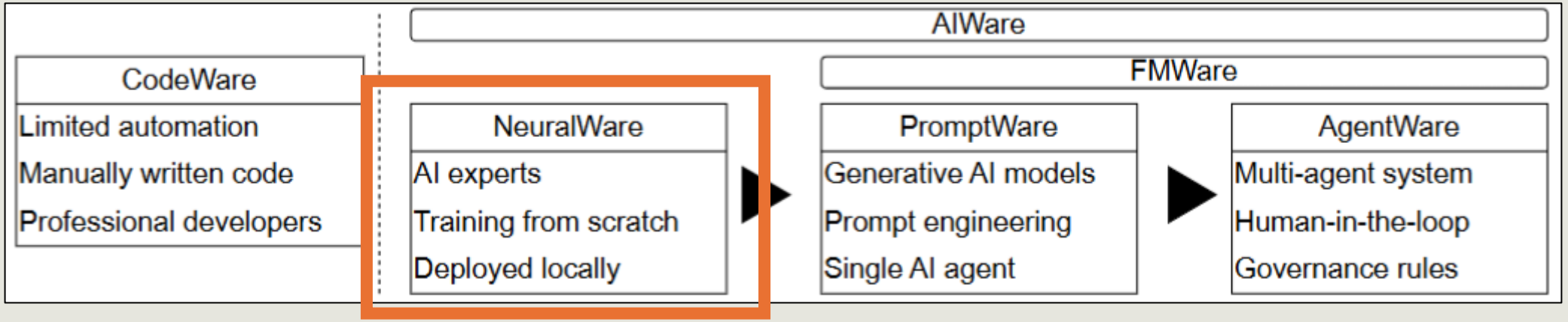
The evolution of software development paradigms



CodeWare development

- Widely adopted paradigm involving professional programmers
- The software lifecycle is driven by traditional phases, i.e., requirement elicitation, design, implementation, testing, deployment, and maintenance, carried out mostly manually

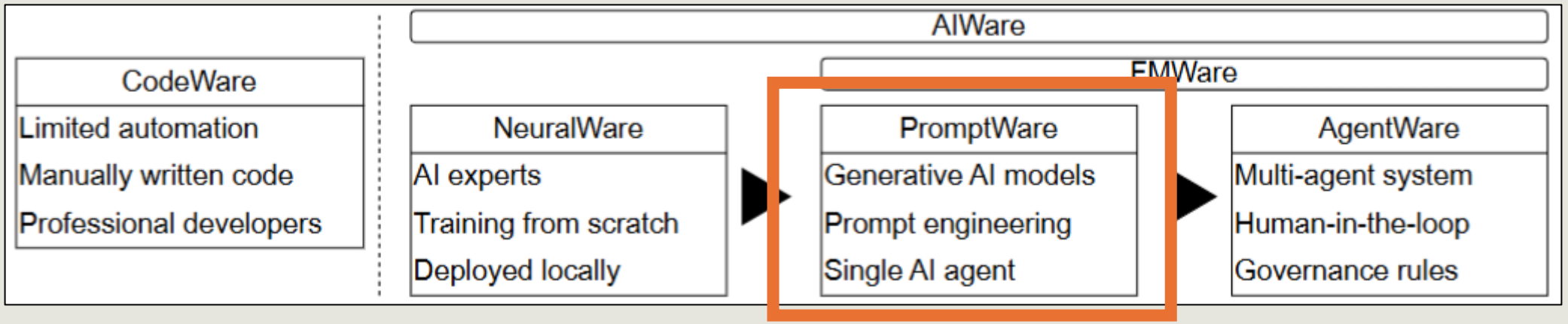
The evolution of software development paradigms



NeuralWare development

- It involves AI experts to drive the development of a new generation of software
- Construction of a new set of assets (e.g., datasets and models in a significantly more iterative and experiment-driven lifecycle).

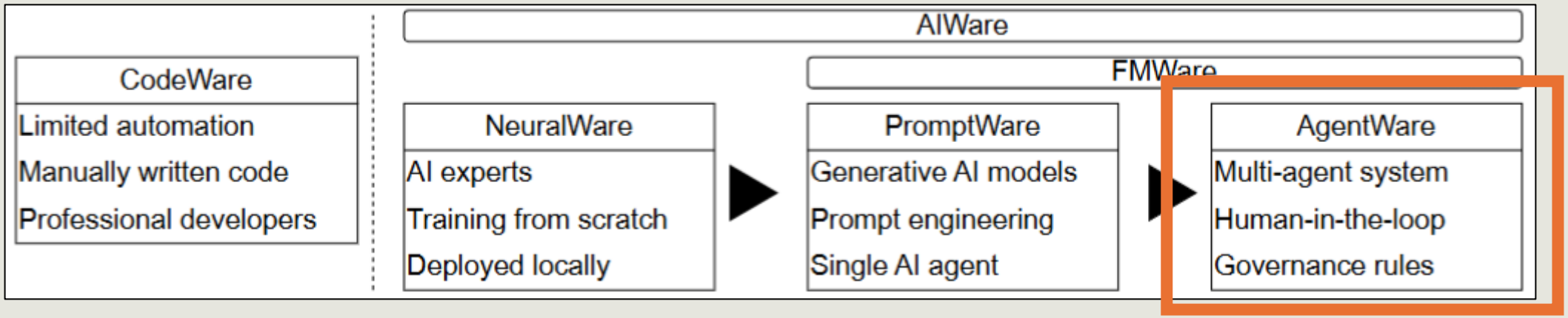
The evolution of software development paradigms



PromptWare development

- Transformer architecture and the launch of ChatGPT
- Prompts enable the software development without deep programming and AI skills
- Improvement of prompting processes (e.g., zero-shot, few-shot, and chain-of-thought)

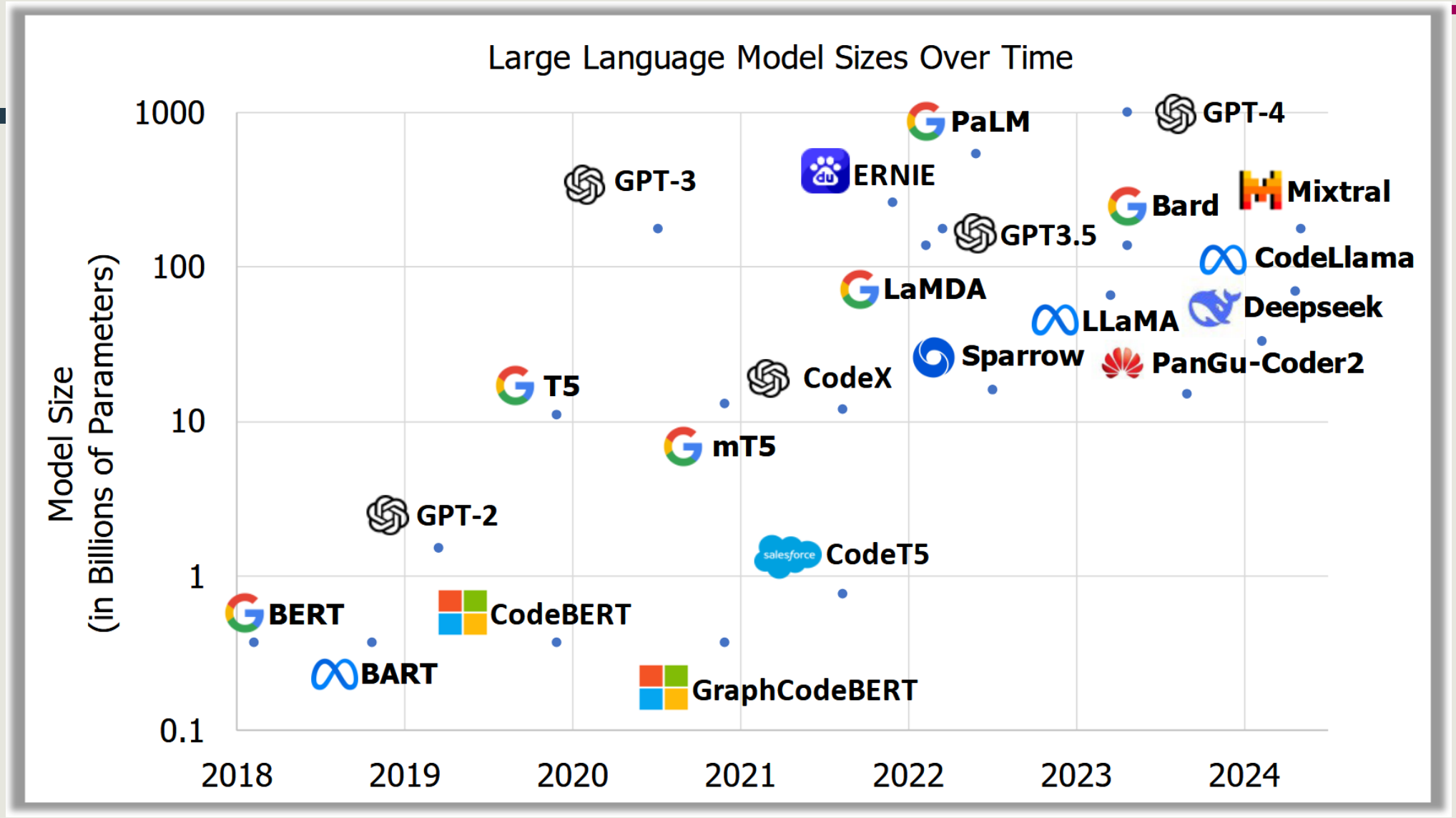
The evolution of software development paradigms



AgentWare development

- Multiple foundation model (FM) agents collaborate to solve tasks by reasoning, decision-making, and interacting with other agents or humans

Large Language Models for SE



David Lo. *Charting New Frontiers: Limits, Threats, and Ecosystems of LLMs in Software Engineering*, Invited talk at ESEM 2024, Barcelona, Spain

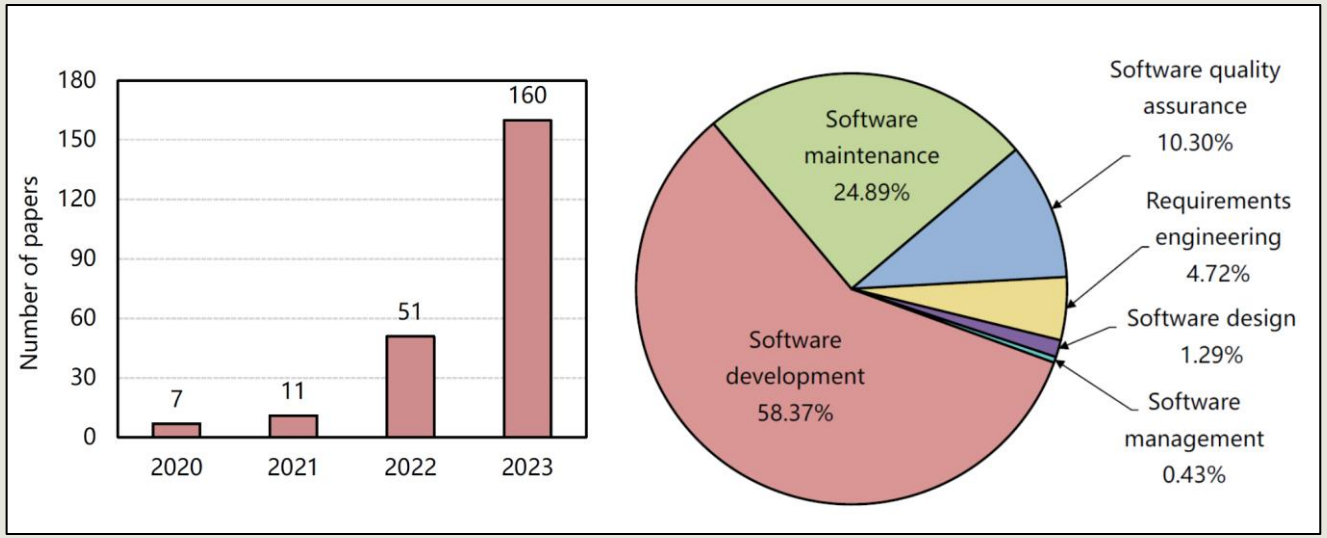
Large Language Models for Software Engineering: A Systematic Literature Review

XINYI HOU and YANJIE ZHAO, Huazhong University of Science and Technology, Wuhan, China
YUE LIU, Monash University, Melbourne, Australia
ZHOU YANG, Singapore Management University, Singapore
KAILONG WANG, Huazhong University of Science and Technology, Wuhan, China
LI LI, Beihang University, Beijing, China
XIAPU LUO, The Hong Kong Polytechnic University, Hong Kong, China
DAVID LO, Singapore Management University, Singapore
JOHN GRUNDY, Monash University, Melbourne, Australia
HAOYU WANG, Huazhong University of Science and Technology, Wuhan, China

Large Language Models (LLMs) have significantly impacted numerous domains, including Software Engineering (SE). Many recent publications have explored LLMs applied to various SE tasks. Nevertheless, a comprehensive understanding of the application, effects, and possible limitations of LLMs on SE is still in its early stages. To bridge this gap, we conducted a Systematic Literature Review (SLR) on LLM4SE, with a particular focus on understanding how LLMs can be exploited to optimize processes and outcomes. We selected and analyzed 395 research articles from January 2017 to January 2024 to answer four key Research Questions (RQs). In RQ1, we categorize different LLMs that have been employed in SE tasks, characterizing their distinctive features and uses. In RQ2, we analyze the methods used in data collection, pre-processing,

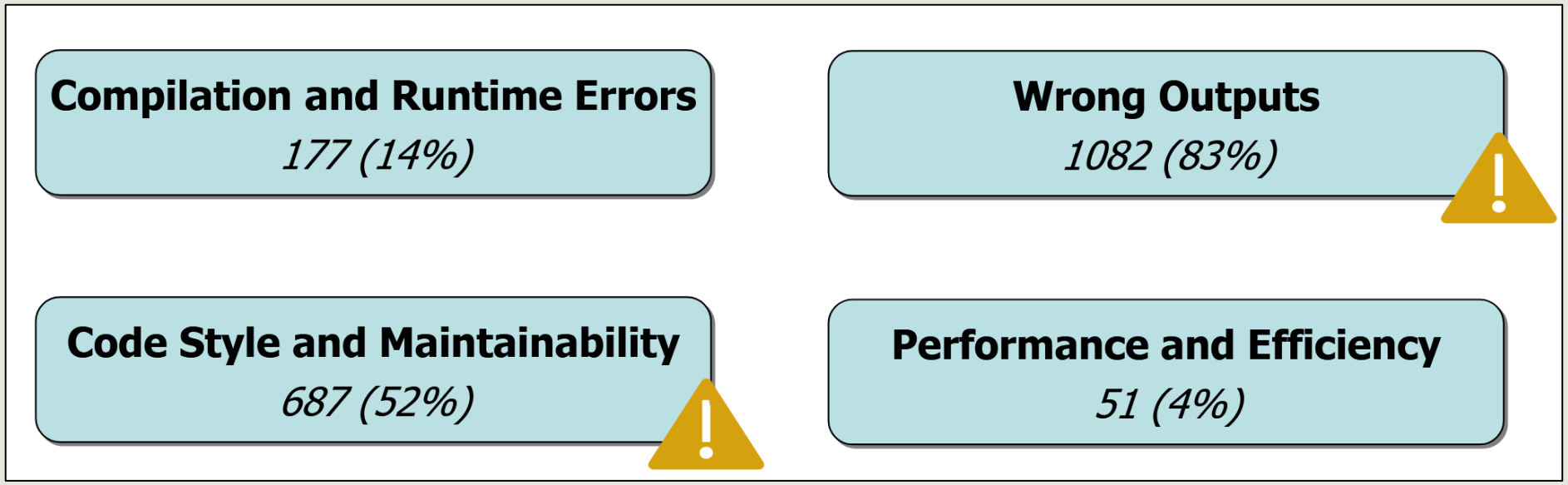
Xinyi Hou and Yanjie Zhao contributed equally to this work.
This research/project is supported in part by the National Natural Science Foundation of China (grant No. 62072046), the Key R&D Program of Hubei Province (2023BAB017, 2023BAB079), the Knowledge Innovation Program of Wuhan-Basic Research, HUST CSE-HongXin Joint Institute for Cyber Security, HUST CSE-FiberHome Joint Institute for Cyber Security and the National Research Foundation, under its Investigatorship Grant (NRF-NRFI08-2022-0002). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore. J. Grundy is supported by ARC Laureate Fellowship FL190100035.
Authors' Contact Information: Xinyi Hou, Hubei Key Laboratory of Distributed System Security, Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan, China; e-mail: xinyihou@hust.edu.cn; Yanjie Zhao, Hubei Key Laboratory of Distributed System Security, Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan, China; e-mail: yanjie_zhao@hust.edu.cn; Yue Liu, Monash University, Melbourne, Australia; e-mail: yue.liu1@monash.edu; Zhou Yang, Singapore Management University, Singapore; e-mail: zyang@smu.edu.sg; Kailong Wang, Hubei Key Laboratory of Distributed System Security, Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan, China; e-mail: wangkl@hust.edu.cn; Li Li, Beihang University, Beijing, China; e-mail: lilicoding@ieee.org; Xiapu Luo, The Hong Kong Polytechnic University, Hong Kong, China; e-mail: csxluo@comp.polyu.edu.hk; David Lo, Singapore Management University, Singapore; e-mail: davidlo@smu.edu.sg; John Grundy, Monash University, Melbourne, Australia; e-mail: John.Grundy@monash.edu; Haoyu Wang (corresponding author), Hubei Key Laboratory of Distributed System Security, Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan, China; e-mail: haoyuwang@hust.edu.cn.

ACM 1557-7392/2024/12-ART220
https://doi.org/10.1145/3695988



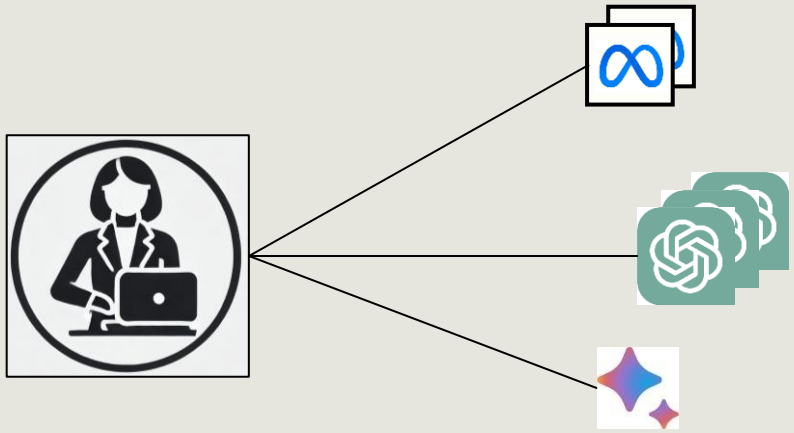
Some challenges

- **In code generation**
  - Hallucination risks lead to syntactically or semantically incorrect code
  - Difficulty in assessing code correctness due to non-deterministic outputs



David Lo. *Charting New Frontiers: Limits, Threats, and Ecosystems of LLMs in Software Engineering*, Invited talk at ESEM 2024, Barcelona, Spain

Some challenges in using LLMs



Diversity: There is diversity in terms of language models, capabilities, and use cases. Developers must navigate this diversity to find the most suitable AI agent for their software engineering tasks.

Lack of Standardized Metadata: Standardized metadata is crucial for understanding the capabilities and limitations of LLMs.

Lack of guidance: Users may need guidance on choosing the most appropriate LLMs for specific software engineering challenges.

Dynamic Nature of AI Landscape: Staying updated on the latest LLM advancements, understanding model capabilities, and assessing their relevance to software tasks is challenging.

LLM-based Multi-Agent Systems

LLM-Based Multi-Agent Systems for Software Engineering: Literature Review, Vision and the Road Ahead

JUNDA HE, Singapore Management University, Singapore
CHRISTOPH TREUDE, Singapore Management University, Singapore
DAVID LO, Singapore Management University, Singapore

Integrating Large Language Models (LLMs) into autonomous agents marks a significant shift in the research landscape by offering cognitive abilities that are competitive with human planning and reasoning. This paper explores the transformative potential of integrating Large Language Models into Multi-Agent (LMA) systems for addressing complex challenges in software engineering (SE). By leveraging the collaborative and specialized abilities of multiple agents, LMA systems enable autonomous problem-solving, improve robustness, and provide scalable solutions for managing the complexity of real-world software projects. In this paper, we conduct a systematic review of recent primary studies to map the current landscape of LMA applications across various stages of the software development lifecycle (SDLC). To illustrate current capabilities and limitations, we perform two case studies to demonstrate the effectiveness of state-of-the-art LMA frameworks. Additionally, we identify critical research gaps and propose a comprehensive research agenda focused on enhancing individual agent capabilities and optimizing agent synergy. Our work outlines a forward-looking vision for developing fully autonomous, scalable, and trustworthy LMA systems, laying the foundation for the evolution of Software Engineering 2.0.

Additional Key Words and Phrases: Large Language Models, Autonomous Agents, Multi-Agent Systems, Software Engineering

ACM Reference Format:
Junda He, Christoph Treude, and David Lo. 2024. LLM-Based Multi-Agent Systems for Software Engineering: Literature Review, Vision and the Road Ahead. 1, 1 (December 2024), 30 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Autonomous agents, defined as intelligent entities that autonomously perform specific tasks through environmental perception, strategic self-planning, and action execution [6, 36, 98], have emerged as a rapidly expanding research field since the 1990s [93]. Despite initial advancements, these early iterations often lack the sophistication of human intelligence [127]. However, the recent advent of Large Language Models (LLMs) [65] has marked a turning point. This LLM breakthrough has demonstrated cognitive abilities nearing human levels in planning and reasoning [3, 65], which aligns with the expectations for autonomous agents. As a result, there is an increased research interest in integrating LLMs at the core of autonomous agents [90, 134, 148] (for short, we refer to them as *LLM-based* agents in this paper).

Authors' addresses: Junda He, jundahe@smu.edu.sg, Singapore Management University, 80 Stamford Rd., 178902, Singapore, Singapore; Christoph Treude, ctreude@smu.edu.sg, Singapore Management University, 80 Stamford Rd., 178902, Singapore, Singapore; David Lo, davidlo@smu.edu.sg, Singapore Management University, 80 Stamford Rd., 178902, Singapore, Singapore.

XXXX-XXXX/2024/12-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Small Language Models are the Future of Agentic AI

Peter Belcak[1] Greg Heinrich[1] Shizhe Diao[1] Yonggan Fu[1] Xin Dong[1]
Saurav Muralidharan[1] Yingyan Celine Lin[1,2] Pavlo Molchanov[1]
[1]NVIDIA Research [2]Georgia Institute of Technology
agents@nvidia.com

Abstract

Large language models (LLMs) are often praised for exhibiting near-human performance on a wide range of tasks and valued for their ability to hold a general conversation. The rise of agentic AI systems is, however, ushering in a mass of applications in which language models perform a small number of specialized tasks repetitively and with little variation.

Here we lay out the position that small language models (SLMs) are *sufficiently powerful, inherently more suitable, and necessarily more economical for many invocations in agentic systems, and are therefore the future of agentic AI*. Our argumentation is grounded in the current level of capabilities exhibited by SLMs, the common architectures of agentic systems, and the economy of LM deployment. We further argue that in situations where general-purpose conversational abilities are essential, heterogeneous agentic systems (i.e., agents invoking multiple different models) are the natural choice. We discuss the potential barriers for the adoption of SLMs in agentic systems and outline a general LLM-to-SLM agent conversion algorithm.

Our position, formulated as a value statement, highlights the significance of the operational and economic impact even a partial shift from LLMs to SLMs is to have on the AI agent industry. We aim to stimulate the discussion on the effective use of AI resources and hope to advance the efforts to lower the costs of AI of the present day. Calling for both contributions to and critique of our position, we commit to publishing all such correspondence at research.nvidia.com/labs/lpr/slm-agents.

https://arxiv.org/pdf/2506.02153

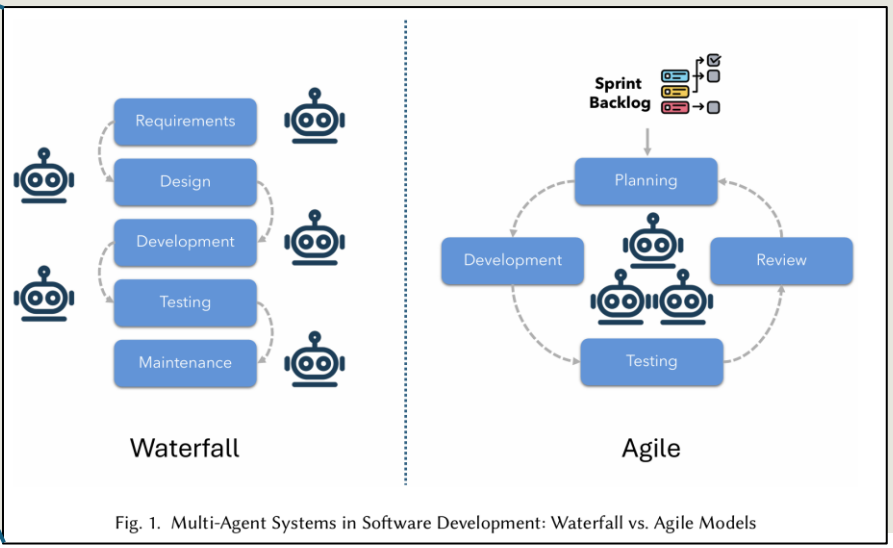


Fig. 1. Multi-Agent Systems in Software Development: Waterfall vs. Agile Models

Agency in AI

- It refers to a system’s ability to act independently to achieve goals
- True agency means an AI system can
  - perceive its environment
  - make decisions
  - act, and adapt over time by learning from interactions and feedback

Agency in the context of LLMs

- Agentic AI involves developing systems that
  - act autonomously
  - understand context
  - adapt to new information, and collaborate with humans to solve complex challenges

- These AI agents leverage LLMs
  - to process information
  - generate responses
  - and execute tasks based on defined objectives

Multi-Agent Systems (MAS)

In many cases, we will want to use multiple agents together

Multiple agents operate in a shared environment, working in a cooperative way, towards a common goal

AI agents are becoming increasingly adept at handling complex use cases

They work best with a good old human in the loop

=> At least for the time being

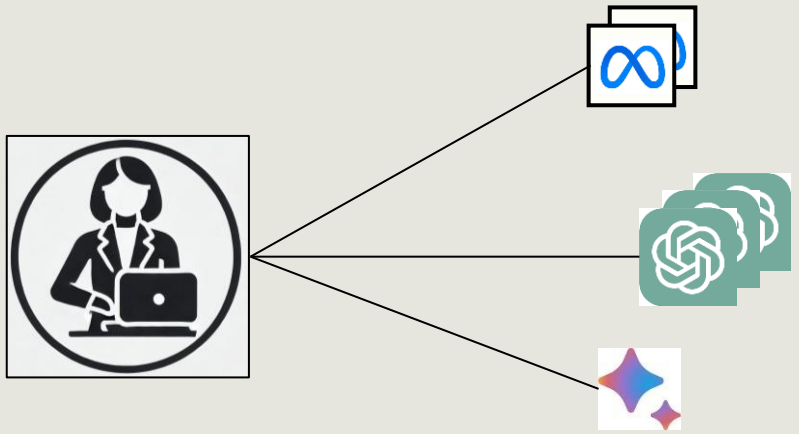
The MOSAICO project

https://mosaico-project.eu/

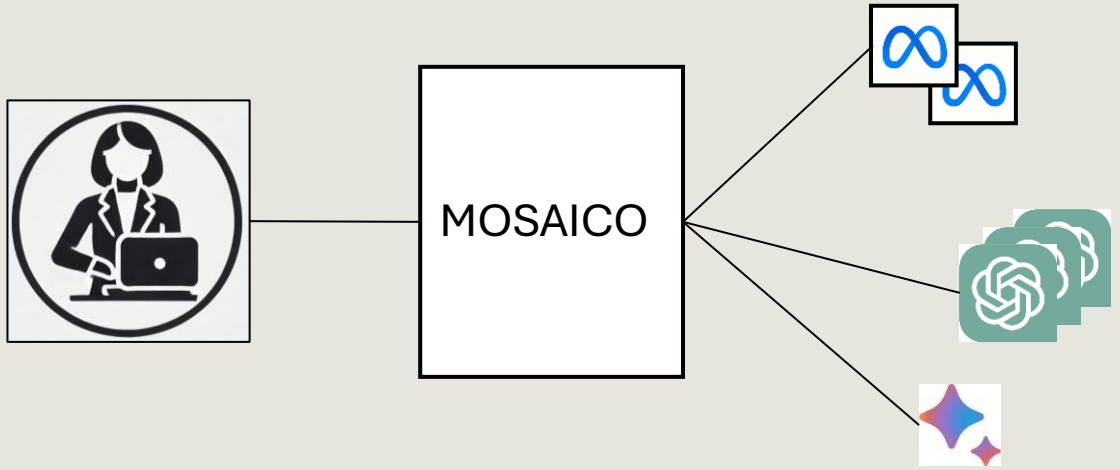
The MOSAICO project in a nutshell

- Acronym: **M**anagement, **O**rchestration and **S**upervision of **AI**-agent **CO**mmunities
- Key objective: Enabling and studying the cooperation of a (possibly large) set of software agents based on Large-Language Models (LLMs), to increase the reliability of LLMs in software engineering
- Project type: Horizon Europe - Research and Innovation Action
- Consortium:
  - Coordinator: IMT Atlantique
  - 4 academic partners
  - 8 industrial partners
- Funding: 5.2 M€
- Dates: January 2025 - December 2027

The envisioned MOSAICO approach

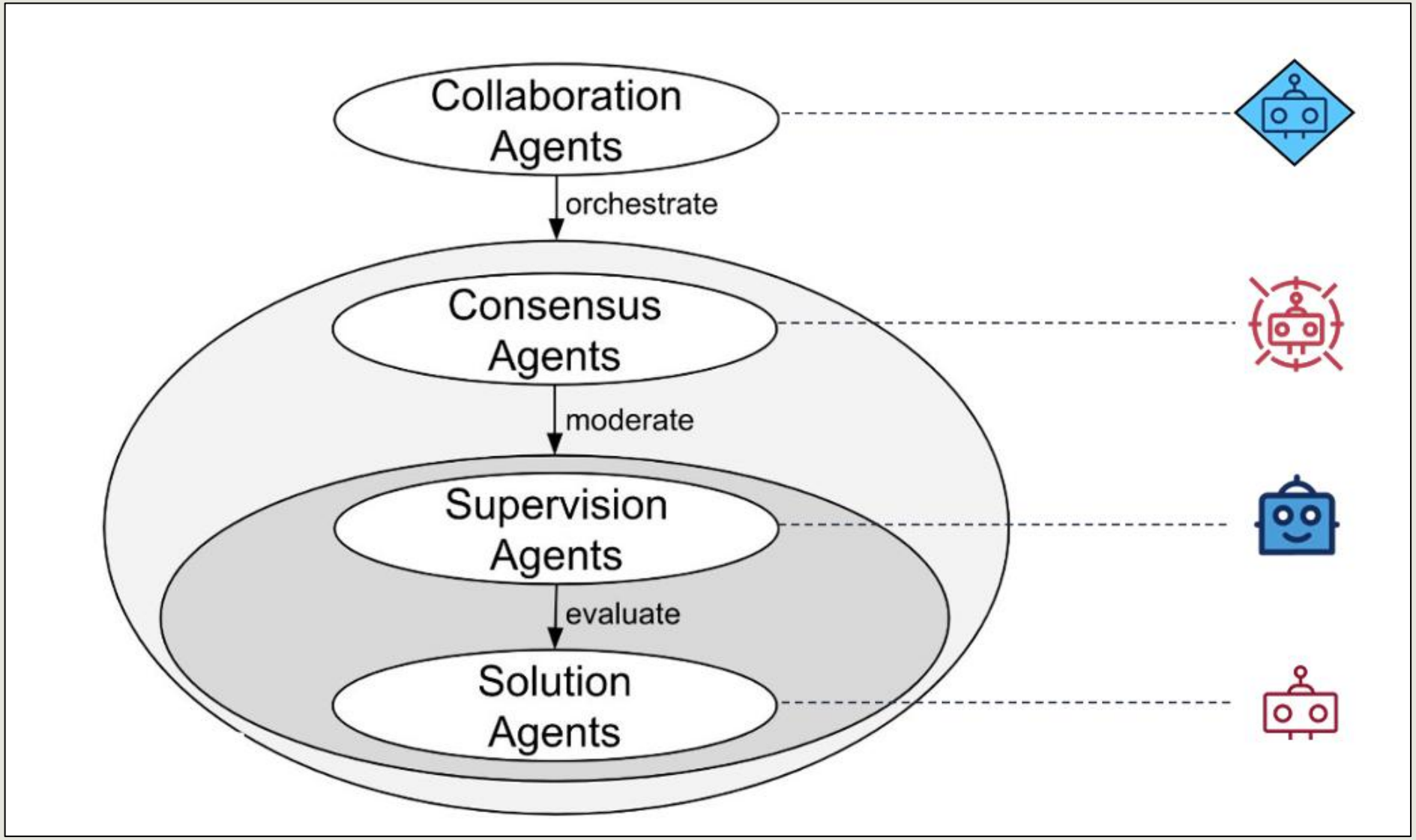


The envisioned MOSAICO approach



MOSAICO platform, handling **communication**, **orchestration**, **governance**, **quality assessment**, **benchmarking,** and **reuse** of **AI agents**.

MOSAICO agents



A basic example

model: codegen-350M-mono

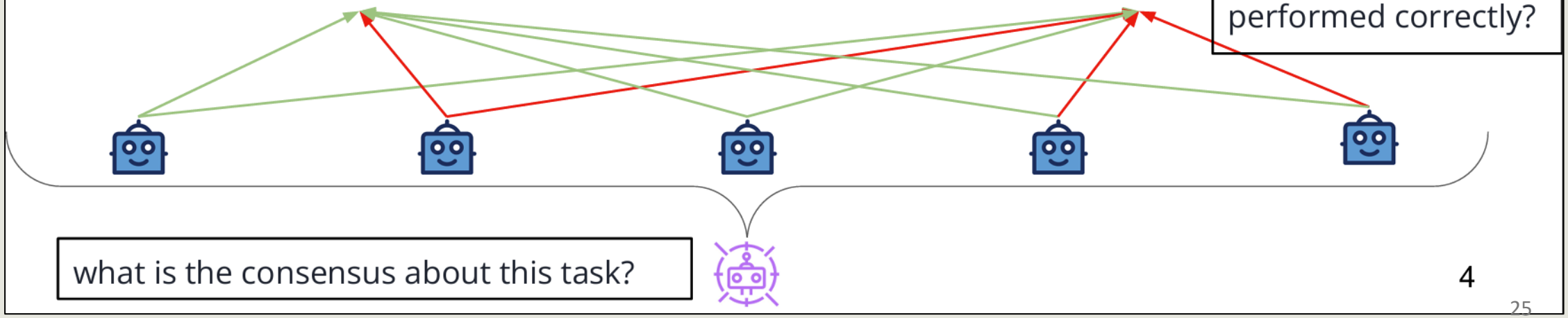
- trained on 71B tokens of Python code

prompt:
```
def fibonacci(n):
```
output:
```
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fibonacci(n-1) + fibonacci(n-2)
```

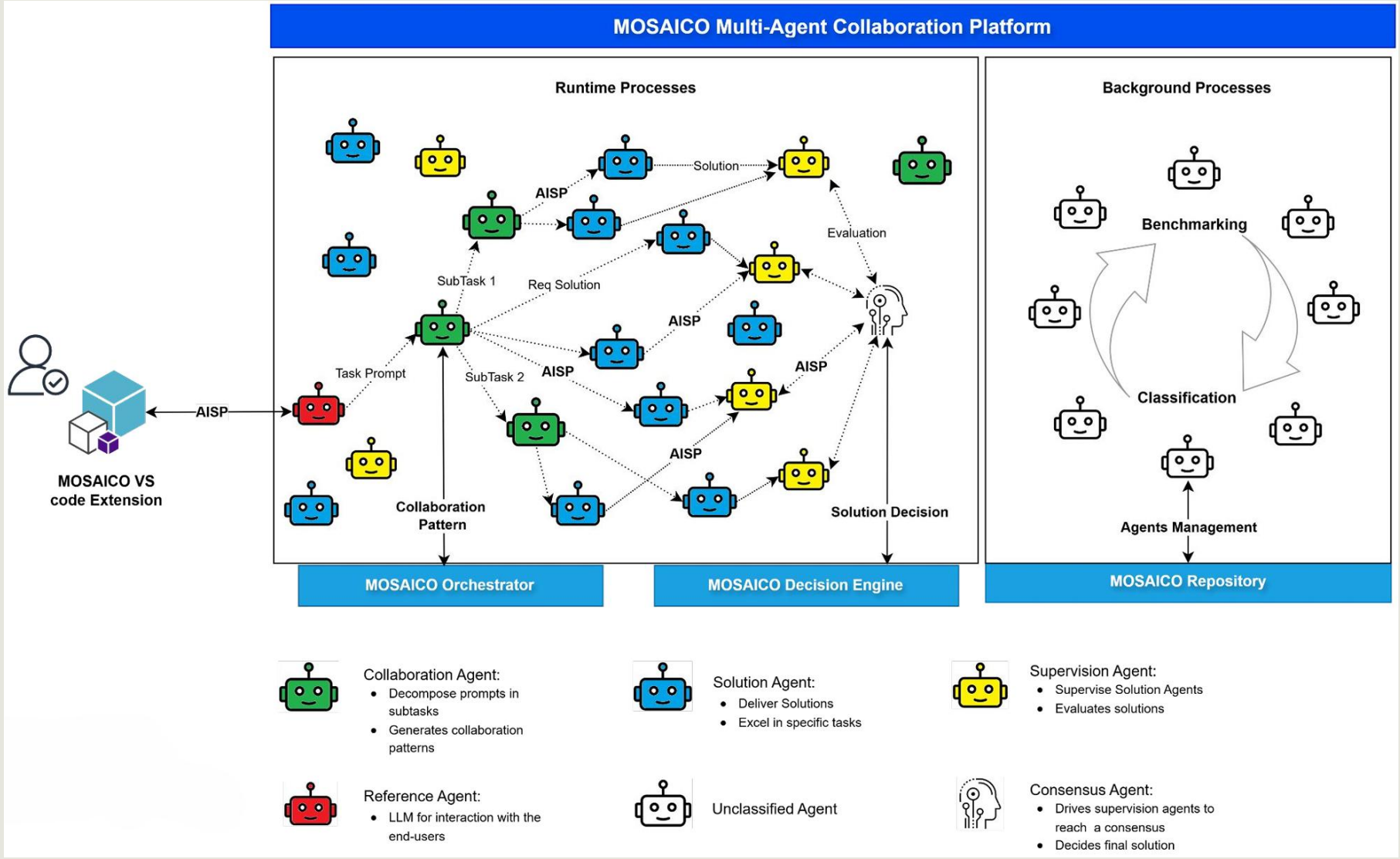
model: gpt-neo-1.3B

- trained on 380B tokens of different kind

prompt:
```
def fibonacci(n):
```
output:
```
    if n == 0:
        return 0
    else:
        return fibonacci(n-1) + fibonacci(n-2)
```



MOSAICO Overview



Key Scientific Objectives (KSO) – 1/2

- **KSO 1: AI-agent server protocol** to allow efficient, reliable, and expandable communication across AI-agent communities
- **KSO 2: Repository of AI agents** managed with respect to supported SE tasks and through a layer driven by QoS KPIs
- **KSO 3: Framework for coordinating collaboration in AI agent communities**, to enable the individual AI agents' decision-making and efficient collaboration
- …

Key Scientific Objectives (KSO) – 2/2

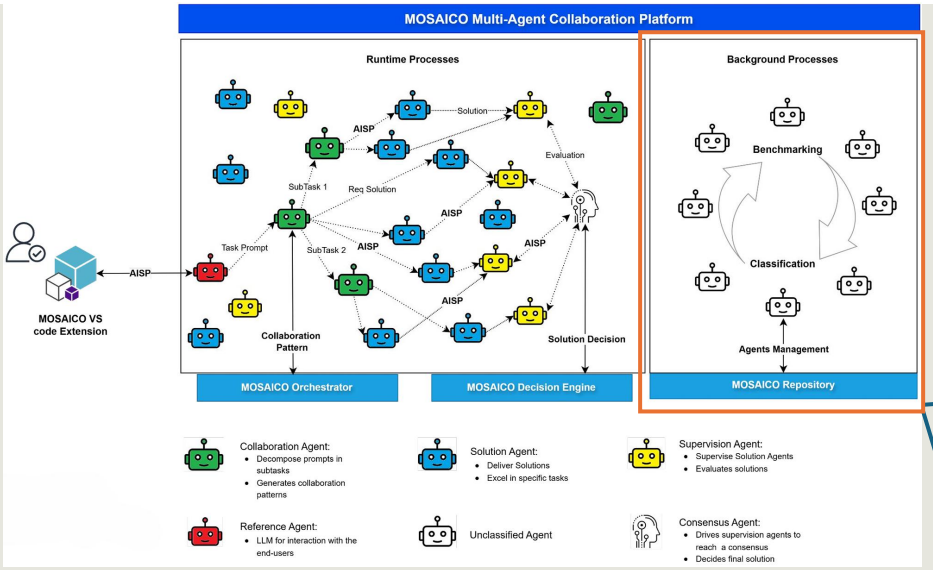
- …
- **KSO 4: Governance language and decision engine** to enact governance policies in the context of collaborative/competitive discussions within communities of Ais
- **KSO 5: Integrated platform** to validate and evaluate 4 different use cases from 4 different application domains

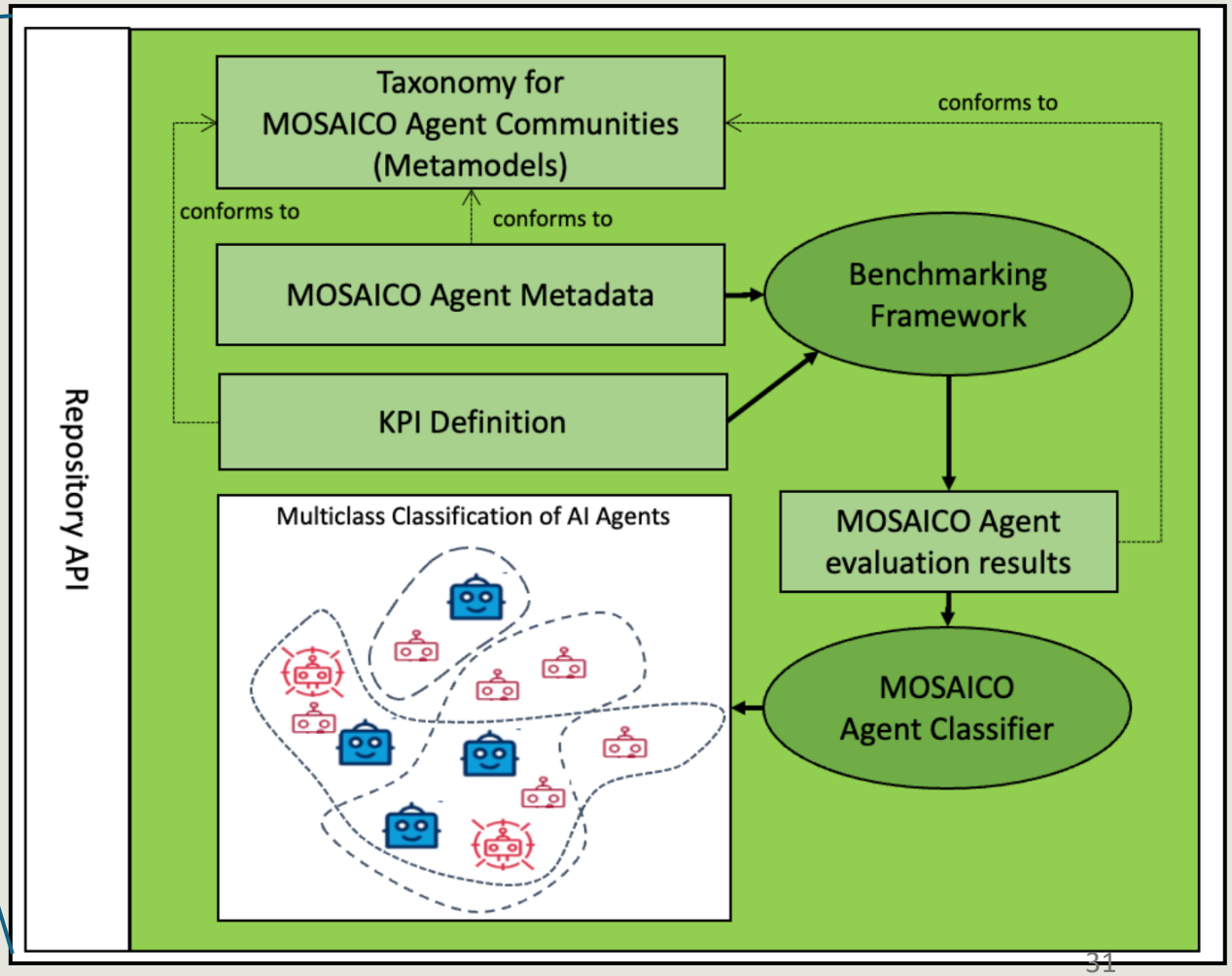
MDE Opportunities

Modeling in the AgentWare Era

- MOSAICO introduces communities of AI agents that must be *specified*, *governed*, and *evaluated*

- Model-Driven Engineering (MDE) provides abstractions and tools to represent:
  - Agent roles and responsibilities
  - Interaction and orchestration patterns
  - Governance and decision policies
  - Quality and benchmarking models

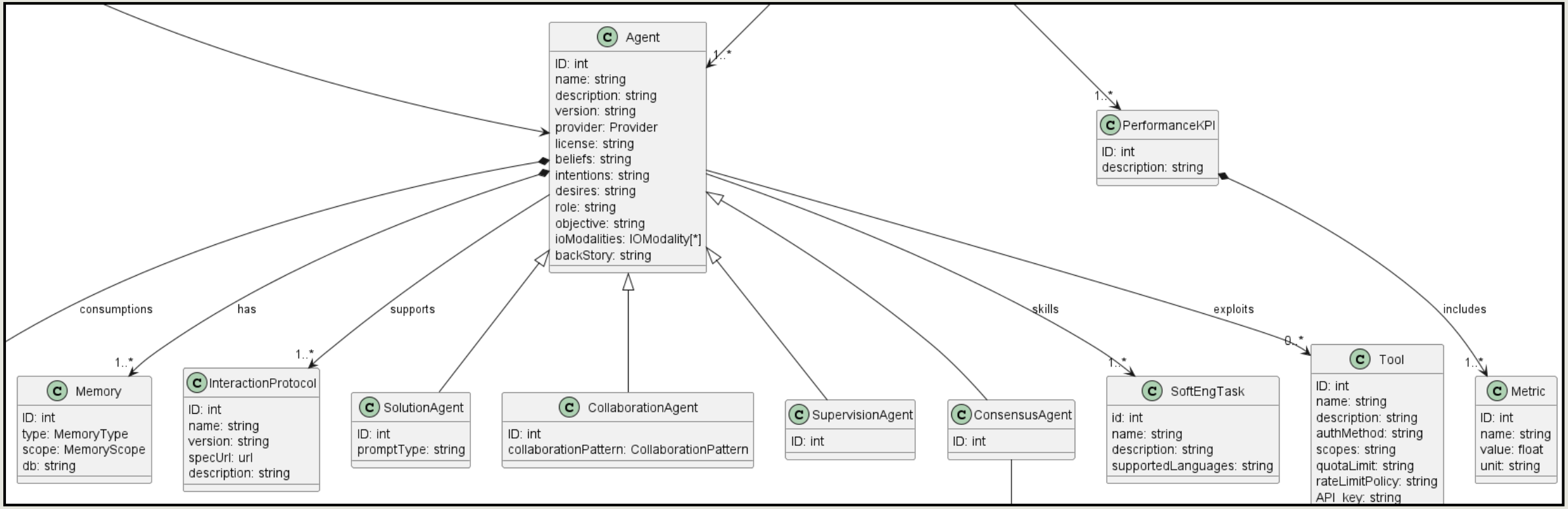




- Quality assessment and categorization for SE tasks
- Metadata for AI capabilities, use cases, and performance metrics

The MOSAICO Agent Taxonomy and Repository

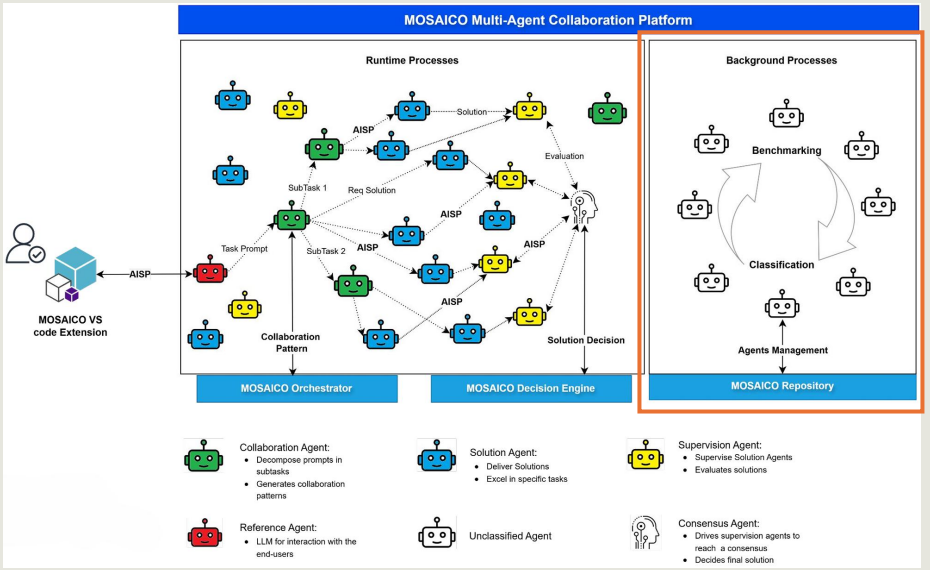
- The taxonomy defines a metamodel capturing the main concepts of *LLM-based agentic systems*
  - Agent, Role, Tool, Benchmark, Policy, TelemetryRecord.



Modeling Quality and Benchmarking

- MOSAICO will define KPI metamodels (accuracy, latency, fairness, reproducibility) to support AI agent benchmarking frameworks

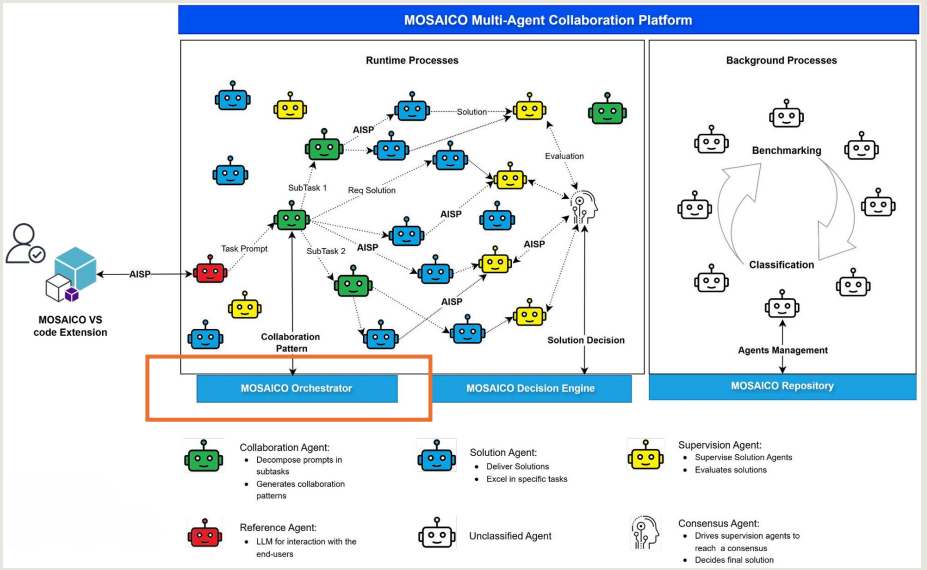
- MDE enables:
  - Definition of measurement models as first-class artifacts
  - Model-based experimentation pipelines comparing agent performance
  - Synchronization of models and metadata to update repository entries dynamically



Modeling Collaboration and Orchestration

- Collaboration modeled through Coordination Patterns

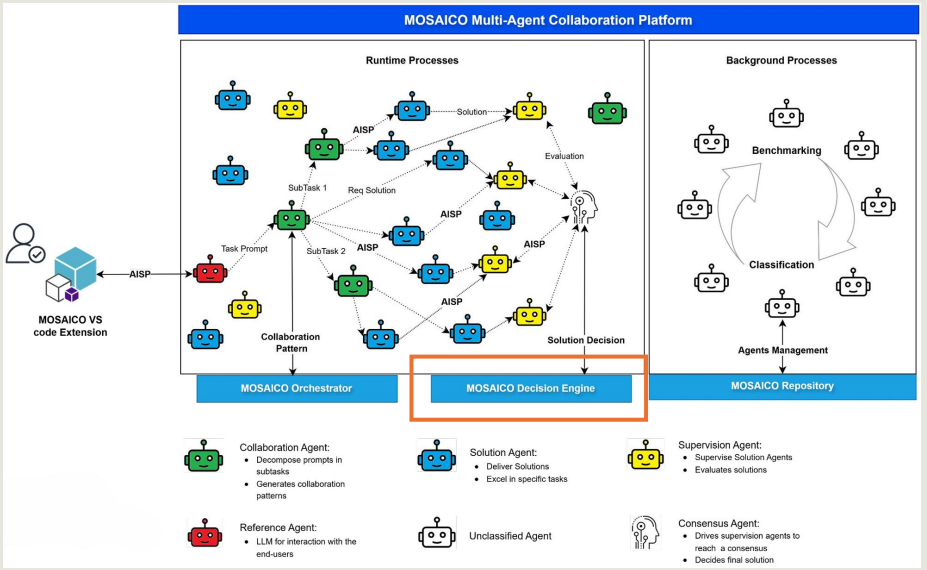
- MDE contributes:
  - Domain-specific coordination languages for multi-agent workflows
  - Model transformations generating executable orchestration graphs



Modeling Governance Policies

- Domain-Specific Language (DSL) for governance

- Abstract syntax metamodel: Actor, Role, Decision, Rule, Constraint
- Concrete syntax (EBNF) + semantics
  → executable by the
  MOSAICO Decision Engine



Opportunities for the MDE Community

- MDE can shape the AgentWare ecosystem by:
  - Creating metamodels and DSLs for agent description, orchestration, and governance
  - Enabling runtime models for continuous monitoring and adaptation
  - Bridging design-time abstractions and execution-time analytics
- MOSAICO offers a living playground for connecting modeling principles to LLM-based agentic systems

Conclusion

Conclusion

- Software development is shifting from

  CodeWare → NeuralWare → PromptWare → AgentWare

- LLMs unlock opportunities in coding, testing, and documentation, but reliability and trust remain open challenges

- The future is multi-agent: collaboration, competition, and supervision among LLM-based agents

Conclusion

- Agents are the new software components; they need repositories, metadata, and benchmarking

- The MOSAICO vision: a holistic ecosystem for agent communities, enabling orchestration, governance, quality assessment, and reuse across SE tasks


STAY TUNED
https://mosaico-project.eu/
MOSAICO