SOFTSTONE 软通动力 | 数字转型新动力

基于 ASDM 的 AI 场景化能力建设 – 2026 年

打造 AI 优先 的软件研发流水线

徐磊







专家介绍

徐磊

- 软通动力 FINSE 软件工程事业部 负责人
- 微软最有价值专家 MVP / 微软区域技术总监 Regional Director
- GitHub Star / Github Copilot 中国区授权服务团队负责人
- 开放原子基金会开源项目 SmartIDE 创始人 / 核心贡献者
- 华为云最有价值专家 MVP
- 资深软件工程 / 敏捷 / 精益 /DevOps 专家
- 资深敏捷教练 / 认证 ScrumMaster
- EXIN DevOps Master/Professional 认证讲师
- 书籍作者 / 译者《专业 SCRUM 基于 Azure DevOps 的敏捷实践》，《云原生应用开发实践》，《基础设施即代码 - 模式与实践》





内容概要

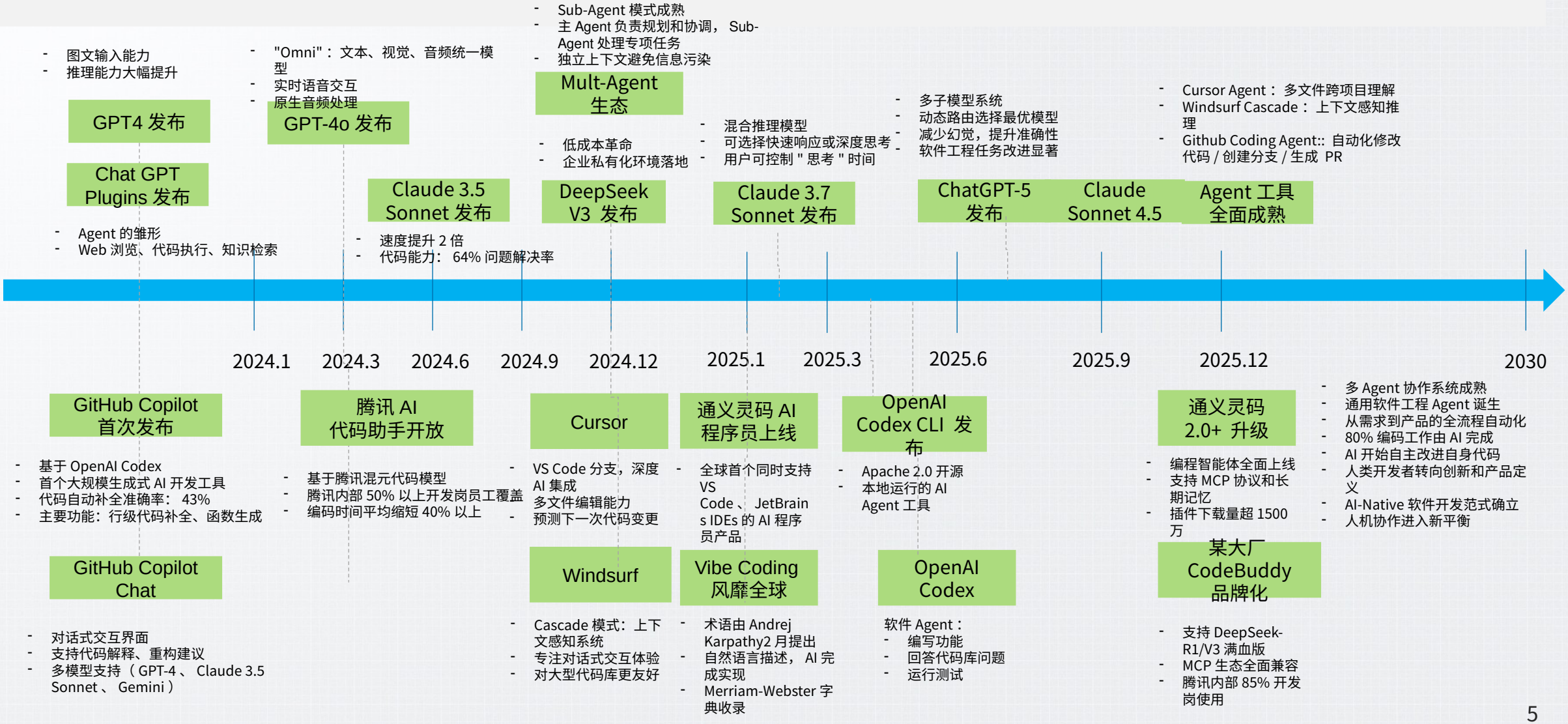
- 行业趋势研究 - 从 AI Coding 到 AI4SE 的快速演进
- 2026 年 AI4SE 能力建设建议 - 围绕 AI 化场景 打造 AI4SE 全流程能力
- ASDM 管理平台 - AI 驱动开发工作流的运营中枢
- AI Coding 超级个体培训

PART 01

行业趋势研究
从 AI Coding 到 AI4SE 的快速演进

AI Coding 行业研究 - 从代码补全到 AI 程序员

当前 AI 编程 Agent 已从“辅助补全”进化到“自主开发”（SWE-bench 82%、可持续工作 30 小时），未来 3 年将实现“端到端产品交付”（95%+ 成功率、多 Agent 协作）。业界需聚焦三大核心：长上下文理解、工具调用能力（MCP 协议）、推理深度控制（可选思考时间）同时专注于垂直领域专精。



AI Coding 行业研究 - Cursor 行业分析报告

Cursor 是由 Anysphere 公司开发的 AI 驱动集成开发环境（IDE），自 2023 年推出以来，已成为 AI 编程工具领域的标杆企业。公司在短时间内实现了快速增长，估值从 2024 年的 26 亿美元飙升至 2025 年的 293 亿美元，展现了强大的市场潜力和投资者信心。

成立时间	创始人	总部	核心产品
2022 年	四位麻省理工学院（MIT）学生	美国	Cursor IDE （基于 VS Code 的 AI 增强编辑器）

产品特点

基于 VS Code，降低学习成本
深度集成 AI，而非简单插件
全流程 AI 辅助，从代码生成到重构
支持多代理（Multi-agent）协作

核心技术

多代理系统
- 多个 AI 代理协作
- 并行处理复杂任务
- 提升代码质量

深度代码理解
- 全代码库索引
- 上下文感知
- 智能重构

自然语言交互
- 对话式编程
- 需求到代码的直接转换
- 代码解释和文档生成

技术架构

基础
-VS Code

AI 模型
- 集成多个 LLM （GPT-4, Claude 等）

协议
- 支持 MCP （Model Context Protocol）

扩展性
- 插件系统

核心成功因素

产品驱动：深度集成 AI，原生 IDE 体验

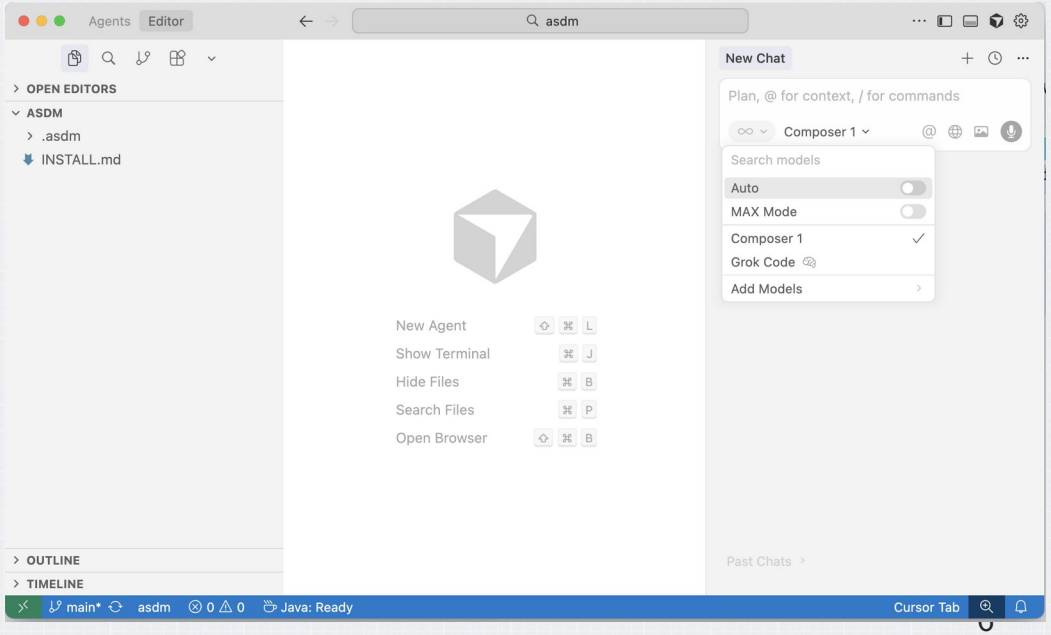
技术领先：多代理系统，深度代码理解

市场定位：专注专业开发者，提升编程上限

快速迭代：持续优化，快速响应市场需求

结论

Cursor 作为 AI 编程工具领域的创新者和领导者，凭借其技术优势、市场表现和资金实力，在短时间内实现了快速增长。公司估值从 26 亿美元增长至 293 亿美元， ARR 达到 3-5 亿美元， NRR 高达 250%，展现了强大的市场潜力和投资者信心。



AI Coding 行业研究 - Claude Code 行业分析报告

Claude Code 是由 Anthropic 公司开发的 AI 辅助编程工具，自 2024 年发布以来，迅速在 AI 辅助编程领域占据主导地位。根据 Gartner 2024 年第一季度报告， Claude Code 以 32% 的市场份额首次登顶，超越 GitHub Copilot （28%）和 Amazon CodeWhisperer （18%），成为行业领导者。用户满意度达到 4.7/5.0，显著高于主要竞争对手。

成立时间	创始人	总部	核心产品
2021 年	前 OpenAI 核心成员	美国	Claude Code （基于命令行的 AI 增强编程工具）

产品特点

命令行原生设计，降低使用门槛，可与任何编辑器 /IDE 配合使用
深度集成 AI，而非简单插件，提供完整的 AI 编程体验
全流程 AI 辅助，从代码生成到审查、重构、优化
超长上下文支持， 200K tokens 上下文窗口，处理大型代码库

核心技术

超长上下文理解
- 200K tokens 上下文窗口
- 全代码库索引与理解
- 深度上下文感知
- 智能代码审查与重构

代码质量优先
- 自动代码审查
- 安全漏洞检测
- 性能优化建议
- 代码规范检查

自然语言交互
- 对话式编程
- 需求到代码的直接转换
- 代码解释和文档生成
- 智能问答与调试辅助

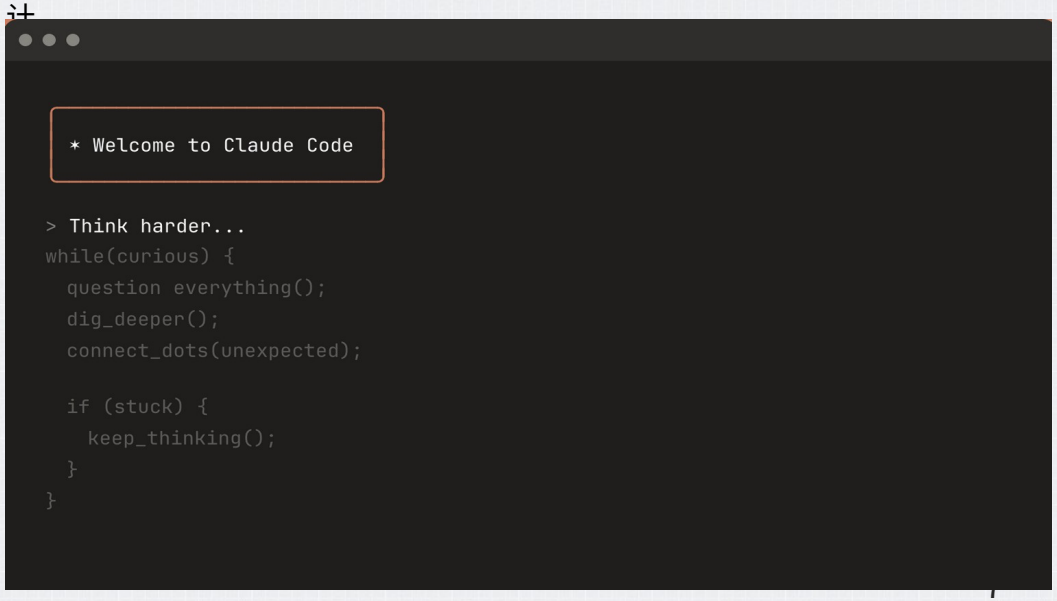
技术架构

基础
- 命令行工具（CLI），轻量级设计

AI 模型
-Claude 3.5 Sonnet / Claude Opus （Anthropic 自研模型）

协议
- 支持 MCP （Model Context Protocol）

扩展性
- 可与任何编辑器 / IDE 集成，支持插件系统



核心成功因素

产品驱动：深度集成 AI，原生命令行体验，灵活可扩展

技术领先： 200K tokens 超长上下文，代码质量优先，安全性保障

市场定位：专注专业开发者，提升代码质量和开发效率

快速迭代：持续优化，快速响应市场需求，企业级支持

结论

Claude Code 是 AI 编程工具领域的创新者和领导者。公司估值从 150 亿美元增长至 300 亿美元。它以其 " 代码质量优先 " 的理念、 200K tokens 超长上下文处理能力和企业级安全标准，在 AI 辅助编程市场中占据领先地位。

AI Coding 行业研究 - 腾讯 Code Buddy 行业分析报告

腾讯 Code Buddy 是由某大厂开发的 AI 驱动编程助手工具。自 2025 年发布以来， Code Buddy 已成为 AI 编程工具领域的标杆企业，在短时间内实现了快速增长。其已广泛应用于政务、金融、工业、医疗等超过 30 个行业，展现出强大的市场潜力和企业级应用价值。

成立时间	创始人	总部	核心产品
2025 年	某大厂团队	中国深圳	Code Buddy （AI 全栈开发工具）

产品特点

三种产品形态 同时支持 IDE 插件、独立 IDE 、 CLI 三种形态，开发者可自由切换，按需选择最适合的工作流。

多智能体协作 覆盖从需求分析到部署的全流程，提升开发效率

双模型架构 支持多模型灵活接入

企业级安全合规 通过等保 2.0 三级

核心成功因素

产品驱动：多智能体协作，全流程 AI 辅助，企业级安全合规

技术领先：双模型混合架构， MCP 协议生态，智能体协作系统

市场定位：专注企业级应用，覆盖 30+ 行业，满足复杂高要求任务

快速迭代：持续优化，快速响应市场需求，深度整合腾讯生态

核心技术

多智能体协作系统
- Plan， Design， Coding， Deploy Agent
- 多文件协同生成
- 自动关联项目依赖关系
- 提升工程理解能力

双模型混合架构
- 根据任务类型动态分配最佳性能
- 技术风险有效对冲
- 支持多模型灵活接入

MCP 协议生态
- 支持 Model Context Protocol 开放生态
- 丰富的预置 MCP Server
- 跨平台、跨工具协同开发
- 企业级扩展能力

结论

Code Buddy 是 AI 编程工具领域的创新者和领导者。它以其多智能体协作系统、双模型混合架构和企业级安全合规能力，在 AI 辅助编程市场中占据重要地位。

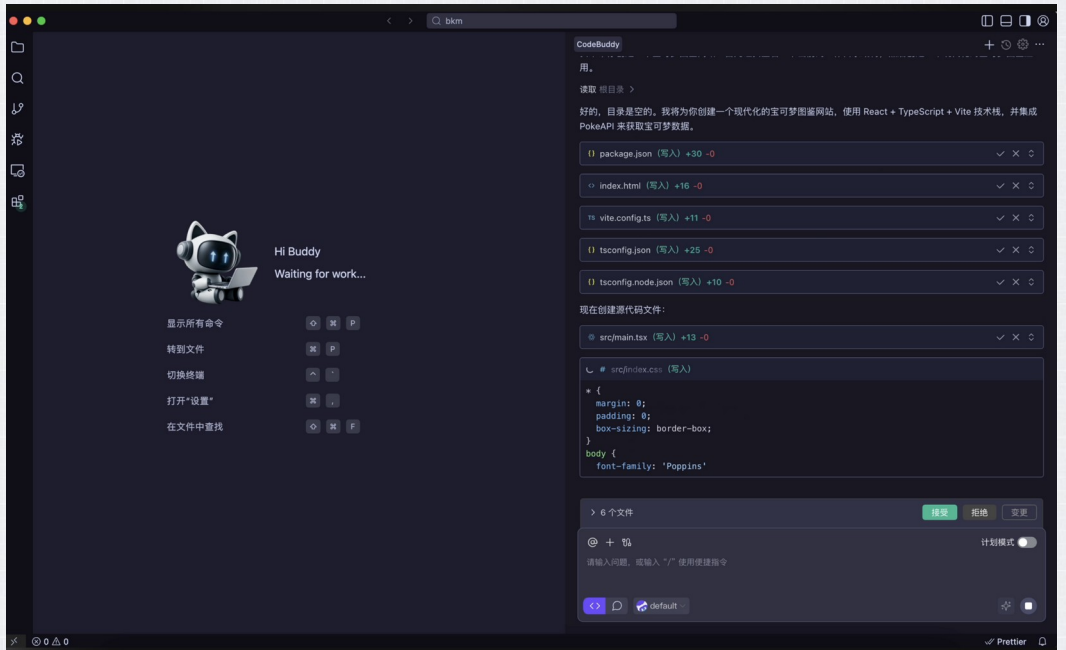
技术架构

基础
- 插件， IDE ， CLI 支持

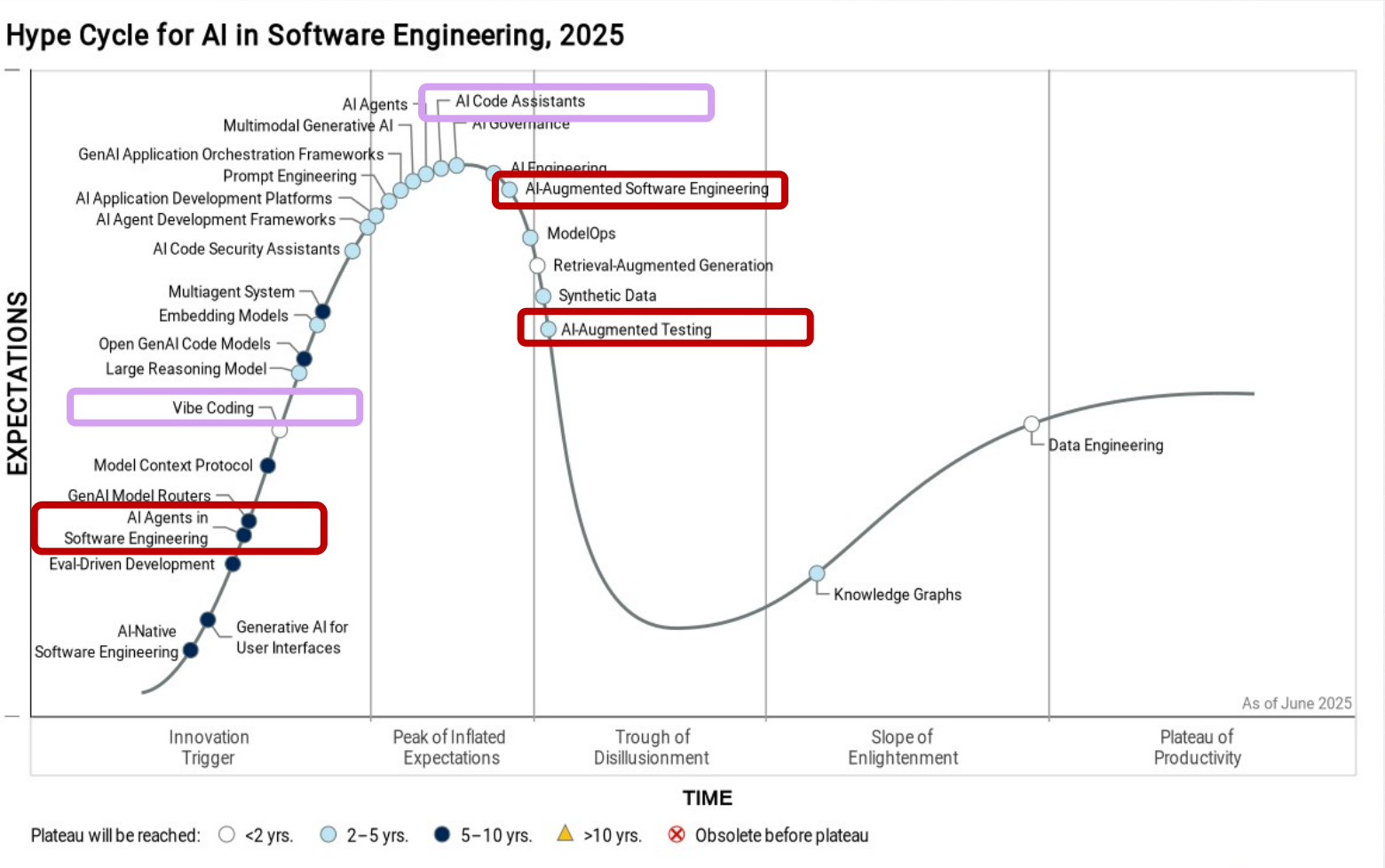
AI 模型
- 混元模型与 DeepSeek 双模型混合架构，支持多模型接入

协议
- 支持 MCP （Model Context Protocol ）开放生态

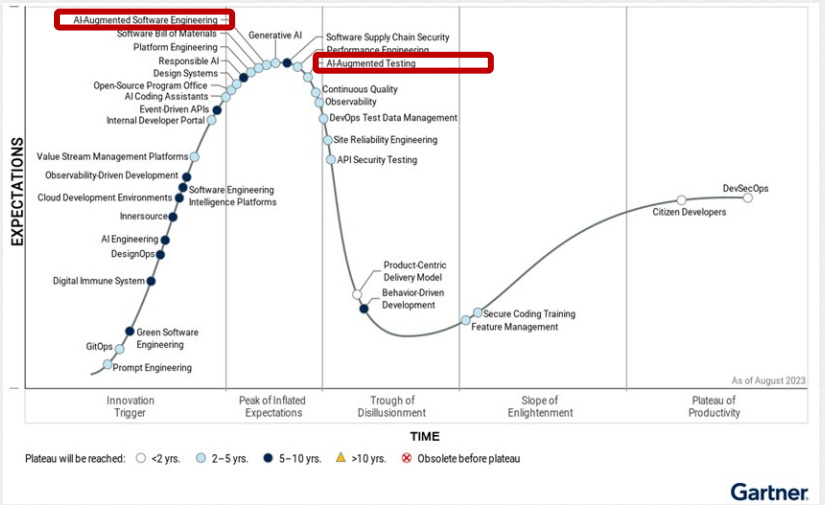
扩展性
- 支持插件、 CLI 、 WebAdmin 、 OpenAPI 、 SDK 等多种形式扩展



AI4SE 研究 - Gartner 的定义和趋势预测



- Gartner 的 SE Hype Cycle 中的定义是 AI-Augmented SE
- 相关概念 AI-Augmented Testing, AI Agents in Software Engineering
- 以上都属于 AI4SE 领域，
- 注意 AI Agent4SE 是 2025 年新出现的概念
- AI-Augmented SE 比较 2023 年的位置出现右移



AI4SE 和 AI Coding 的区别

关联： AI Coding 是 AI4SE 最成熟的细分领域和核心切入点； AI4SE 是在其基础上将 AI 能力延伸至全流程，实现研发体系的整体智能化，二者是**子集与整体、单点应用与体系化升级**的关系。

	AI4SE (AI for Software Engineering)	AI Coding (AI 辅助编码)
核心定位	赋能软件工程**全生命周期**的体系化方法论与实践框架。	聚焦软件**编码单一环节**的 AI 辅助技术 / 工具，是 AI4SE 的子集
覆盖范围	需求、架构、编码、测试、运维、项目管理等全流程。	代码编写、调试、重构、注释、单元测试生成等。
核心目标	提升**整体研发效能**，保障全生命周期质量，优化流程。	提升**编码环节效率**，降低门槛，减少错误，缩短周期。

Vibe Coding vs. Spec Driven Development (SDD)

Vibe Coding (氛围编程)

定义与起源

由 Andrej Karpathy 提出，通过自然语言描述“感觉”， AI 生成代码，强调快速迭代与探索。

核心思想

“让 AI 写代码，人类只管提需求”。拥抱指数级变化，忘记代码细节，专注创意。

适用场景

快速原型验证、个人项目开发、创意探索与试错。

Spec Coding (规格驱动编程)

定义与起源

2025 年下半年兴起的企业级范式，先定结构化规格文档， AI 严格按 Spec 生成并验证。

核心思想

“先定规格，再生成代码，全程校验闭环”。强调工程化、可维护性和团队协作。

适用场景

企业级复杂系统开发、大型团队协作项目、需要长期维护的软件系统。

Vibe Coding vs. Spec Driven Development (SDD)

Vibe Coding (氛围编程)

- 核心输入：自然语言模糊描述，即兴发挥
- 代码质量：可控性低，可用率约 30% (原型级)
- 团队协作：需求对齐难，返工率高 (≥60%)
- 适用场景：原型验证、个人项目、创意探索
- 核心特征：速度快、门槛低，但维护成本高

总结：快 vs 稳

Vibe Coding 的优势在于速度和创造力，适合快速试错；但劣势是质量不可控，维护成本高。

Spec Coding (规格驱动编程)

- 核心输入：结构化 Spec 文档，规划式工程化
- 代码质量：高可控性，可用率 85%+ (生产级)
- 团队协作：完美适配， Spec 即契约，返工率低
- 适用场景：企业系统、团队协作、长期维护
- 核心特征：质量高、风险可控，但前期投入大

总结：工程化首选

Spec Coding 虽然前期投入大，但保证了高可用率和低返工率，是企业级应用和长期维护的最佳实践。

Vibe Coding vs. Spec Driven Development (SDD)

历史与现状

Vibe Coding (2025 年初)

由 Karpathy 提出，以极致速度著称，成为快速原型的代名词。但因代码质量和可维护性问题，在工程化落地中受到质疑。

Spec Coding (2025 年下半年)

由亚马逊、微软等企业推动，旨在解决工程化问题。强调规范与可靠性，已逐渐成为企业级 AI 开发的主流范式。

未来展望

范式融合趋势

非此即彼，而是结合使用。例如先用 Vibe Coding 快速探索原型，再用 Spec Coding 进行工程化落地。

智能化工具进化

工具将支持从 Vibe 对话自动生成 Spec ，或在 Spec 中允许更灵活的动态调整，降低转换成本。

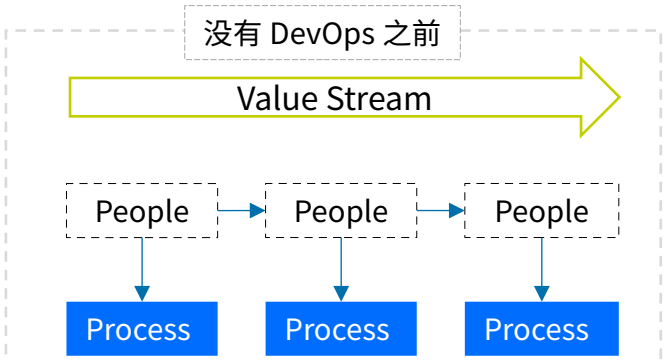
开发者角色转变

开发者将更多地扮演“定义者”和“审核者”的角色，专注于问题定义与结果校验，而非单纯的代码编写。

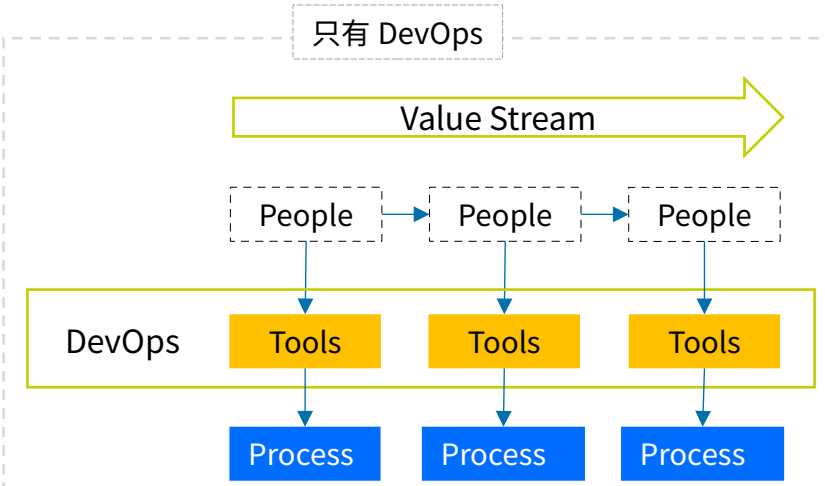
AI4SE 与 DevOps 的关系

SOFTSTONE 软通动力 | 数字转型新动力

软件工程能力 / 成熟度

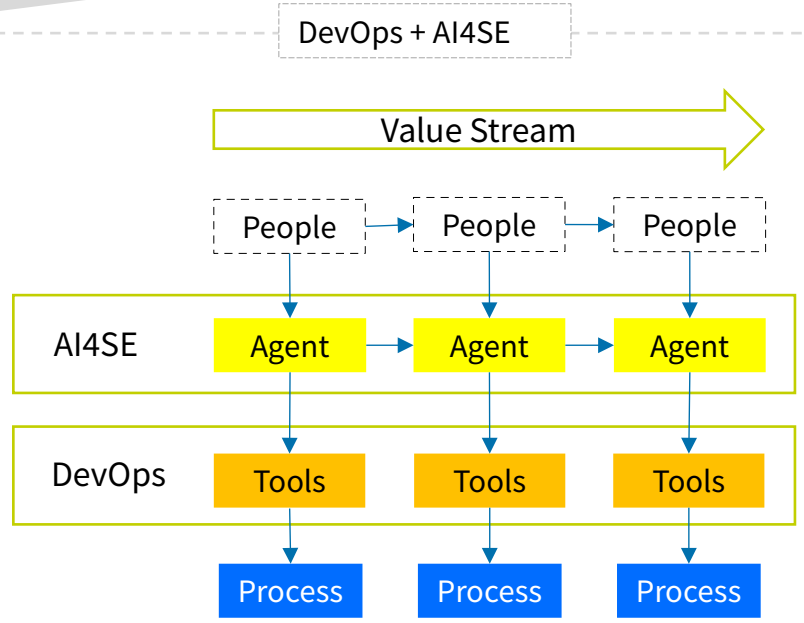


- 依赖人工完成价值流动
- 没有标准的工具支撑
- 没有数据收集能力
- 没有知识积累和传承



- 工具辅助的价值流动
- 构建每个环节的最好领域工具，工具链集成
- 领域数据可采集
- 领域知识被积累

- 人工操作工具和沟通仍然是保证价值流动的关键因素
- 数据统一性和完整性问题
- 事务性工作造成额外开销



- 生成式 AI 和工具为先的价值流动
- 各阶段领域工具紧密链接
- 端到端数据链接开始建立
- 端到端知识积累能力持续提升

- 构建项目端到端认知能力成为可能
- 数据统一性和一致性问题得到缓解
- 事务性工作开销减少

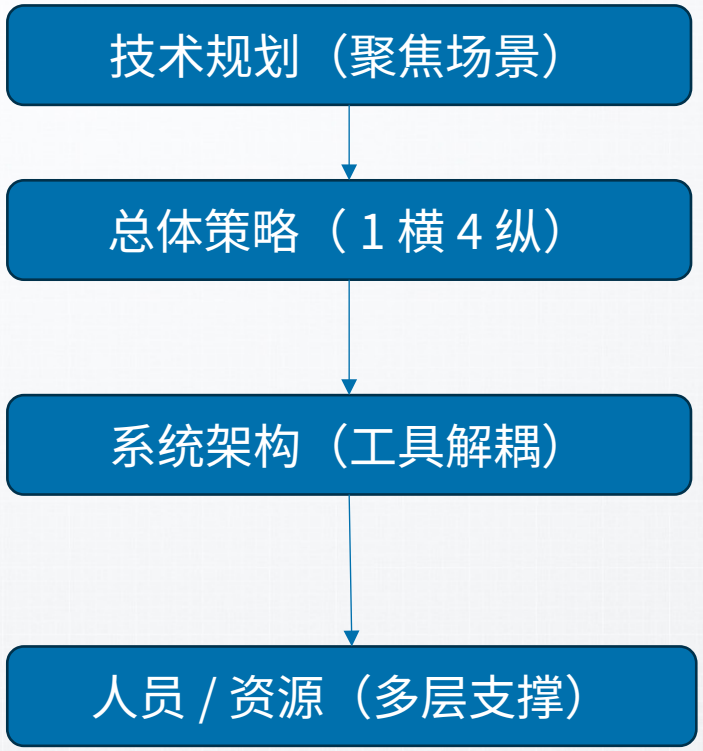
PART 02

2026 年 AI4SE 能力建设建议
围绕 AI 化场景 打造 AI4SE 全流程能力

如果 AI 可以已人类 100 倍的速度生成代码
那么我们只要为 **AI 构建适合的方法 / 工具**
确保 AI 生成的准确性
就可以用 100 倍的速度构建软件

在未来的 5-10 年，开发人员的职责
不再是为 人构建软件 / 工具
而是 为 AI 构建软件 / 工具

2026 年 AI Coding 项目规划



- 依托行业先进商业化产品、集中资源构建 **AI 化场景** 、打通企业内部研发工具链，形成端到端 **全流程 AI 场景化能力**。

- 围绕 **编码、代码评审、代码检查、集成测试** 4 个场景全面铺开通用工具支撑
- 围绕 **试点团队**，招募有意愿，有影响力的内部业务团队，协同打造 端到端 AI4SE 研发流水线

- 基于 ASDM 方法论体系，围绕 **ASDM 平台** 构建 **AI 化场景** 的交付闭环，最大化利用业界层出不穷的最佳实践（ MCP ， Skills, SugAgents, Hooks ）等资源。
- 构造与 **AI Coding 能力** 层结偶的 **AI4SE 能力** 层，厂商解绑，自主可控，工具自由

- **基础设施保障：**提供稳定运行的 AI4SE 基础设施，包括：算力、编码能力大模型、推理服务、 **AI Coding 基础服务、 AI Coding 插件 /IDE/CLI**
- **AI Coding 能力保障：**围绕标准化产品特性，提供培训和运营，以问题清单为抓手，保障 AI4SE 底层能力的稳定运行。
- **ASDM 基础平台开发：**基于 ASDM 标准化产品，对接行业研发工具体系，形成全流程 AI 场景化支撑能力。
- **AI Coding 架构师 / 教练：**围绕 **AI 工具包** 的构建，为业务开发团队提供标准化的 **AI 场景** 构建和导入服务，使能业务团队实现 **全流程 AI 场景化**。

ASDM 方法论和平台 – 打造 AI 优先 的软件工程能力体系

SOFTSTONE 软通动力 | 数字转型新动力

基于 **ASDM (AI 优先系统化研发方法和平台)** 覆盖软件研发全流程，为不同软件研发角色在不同阶段提供针对性的 **AI 化场景** 支撑和提效，为企业提供端到端的交付价值流输出。通过全生态兼容能力最大化厂商合作可能性，为企业实现 AI4SE 全场景赋能提供清晰的落地实施路径和可管理性。

ASDM 方法论体系 – > 从 为人构建流程和工具 全面转向 为 AI 构建流程和工具 -> 改变研发生产关系 -> 获取 10 倍以上的生产力能力提升

AI 优先的软件工程能力体系和平台

标准化交付
产品 + 集成 + 服务

- **ASDM 产品研发与标准化交付**
  企业级规约和标准化业务流程交付
- **DevOps 工具链集成开发服务**
  全面对接企业研发管理工具链
- **AI4SE 最佳实践咨询服务**
  资深 AI 架构师提供最佳实践提炼和导入

场景能力层
研发业务赋能
最佳实践导入
标准化规约模版

AI 化场景 / ASDM 工具包

		任务级代码生成 / 补全		
		代码重构 / 转换 / 升级		
		设计对接 / 一致性保障		CI/CD 流水线创建 / 编排 / 维护
	数据建模和设计	前后端协同开发	规范和安全扫描和修复	日志分析 / 部署策略优化
需求编写辅助	API 接口设计	单测生成 / 覆盖率保障	质量红线检查和修复	事件问题分析 / 代码级定位
实现方案调研和设计	实现方法和详细设计	测试用例生成 / 自动化	代码评审 / 问题和修复建议	流量监控 / 云资源调度
原型设计和搭建	任务分解和计划制定	缺陷定位 / 根因分析	度量指标和仪表盘	用户行为监控 /Insight 生成

核心能力层
修复模型幻觉
确保生成准确性

- **构建场景化认知能力 - 提供 AI 友好的上下文**
- **构建闭环执行能力 - 提供 AI 友好的工具**
- **构建自我迭代能力 – 保持项目状态同步**

研发全流程覆盖

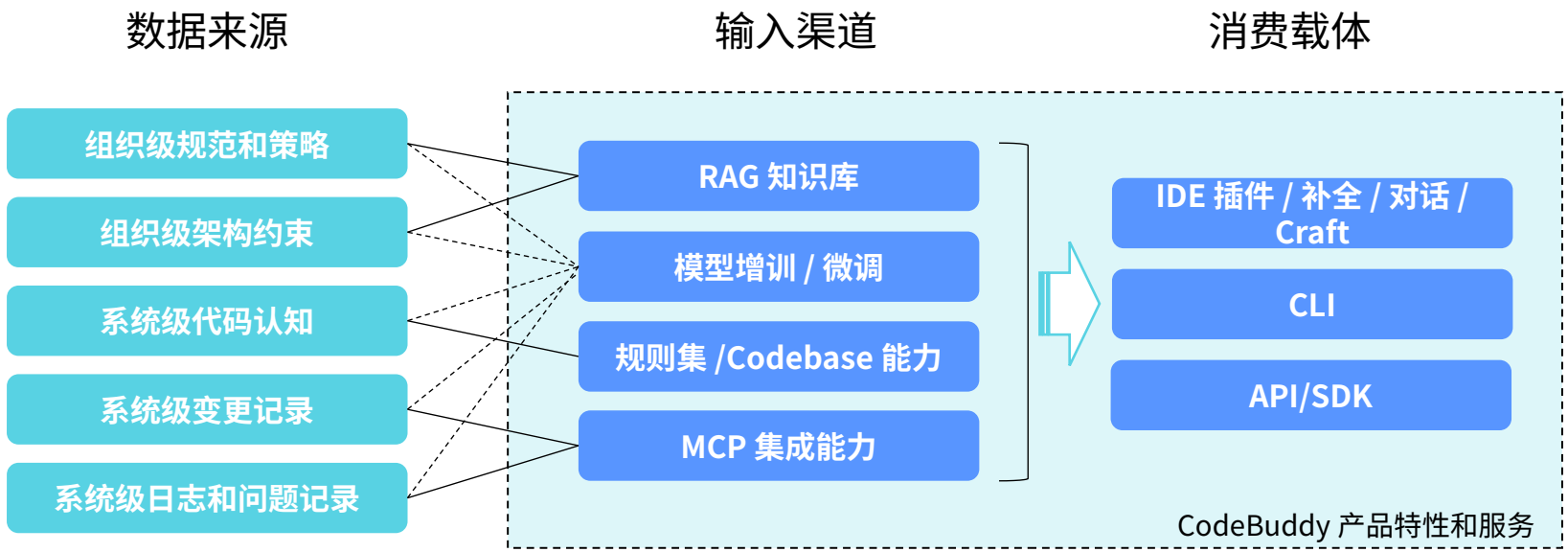
- **需求与设计**
  认知 / 提炼 / 调研 / 原型 / 方案
- **架构与规划**
  数据建模 /API/ 模式推荐
- **编码和测试**
  代码生成 / 重构 / 转换 / 评审
- **QA 与安全**
  红线 / 左移 / 自动修复
- **DevOps 与持续交付**
  自动化 / 日志 / 根因 / 部署 / 行为

全生态兼容合作能力

- **AI 编码能力层 - 同时支持 国内 AI 供应商（腾讯 / 阿里 / 字节 / 百度 / 华为）和 国际化供应商（微软 / 谷歌 /OpenAI/ Anthropic/AWS ）**
- **AI 模型能力层 – 同时支持各类 国产信创算力（华为 / 阿里 / 百度 / 海光）和 国际化算力 （英伟达）以及 API 接入能力 （阿里 / 字节 / 腾讯 / 智普 /Kimi/Azure/AWS 等）**

核心实践： ASDM 基础能力 – 构建场景化认知能力 - 提供 AI 友好的上下文

项目级认知 是为 AI/ 人类 了解产品 / 项目提供充分，完整，准确的上下文信息的基础能力，通过 AI 与人的互动完成对上下文的采集，整理和版本管理，借助 agentic 引擎 / 规则集 / 知识库 /codebase/ 模型微调 /MCP 等能力对项目知识体系进行维护，并在智能体需要的时候有效的注入上下文，拉齐 AI 和人类对项目的认知水平，确保生成结果的准确性。



上下文工程（Context Engineering)

项目级认知 是 CodeBuddy 解决方案中依据 **上下文工程** 实践方法，结合 Codebuddy 各项特性为企业提供的面向软件工程全流程（SDLC）的解决方案。

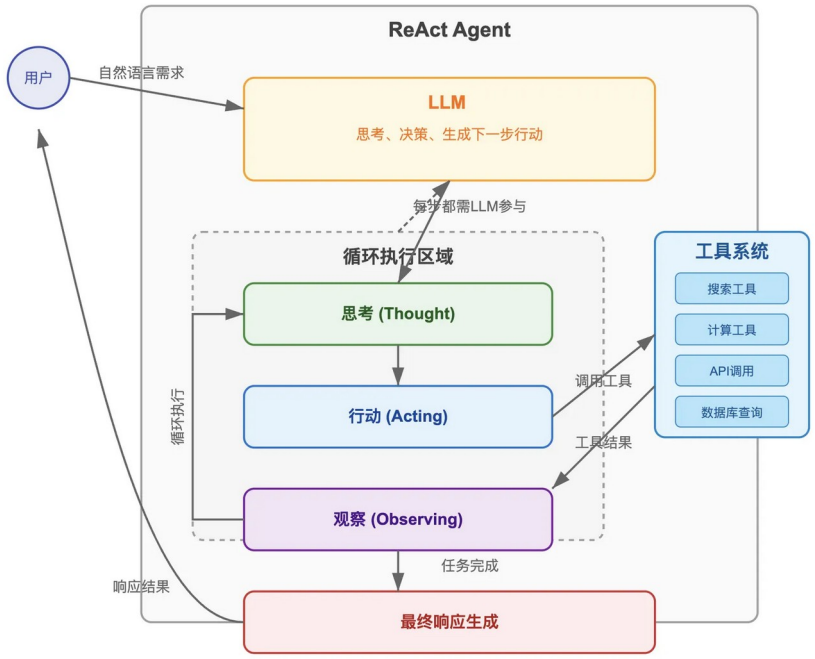
上下文工程更加强调构建完整的上下文系统，而不仅仅提供零散而且不易管理的提示词。企业需要通过系统化的工程能力构建，为系统开发提供充分、完整、准确的上下文信息，并提供有效的端到端上下文数据的生命周期管理能力。

目标场景

- 需求阶段：增强 AI 对需求理解的完整性和纬度，辅助需求理解，提炼，原型，场景和解决方案设计
- 架构阶段：辅助完成数据和系统建模， API 设计，架构选型，技术方案验证
- 规划阶段：辅助完成任务拆分和详细设计
- 编码阶段：提升生成代码的适配性、组织规范 / 策略注入、提升生成效率和质量
- 测试阶段：确保测试用例完整性、企业个性化适配性和各类规范约束的充分引入
- QA 与安全：各类组织红线 / 检查点 / 安全问题的及时发现和规则集维护
- DevOps 与持续交付：构建符合组织要求的流水线结构、日志 / 根因分析中结合企业规范，用户行为分析结果直接提出修改建议

核心实践： ASDM 基础能力 – 构建闭环执行能力 - 提供 AI 友好的工具

自主任务执行 是围绕 AI 的 ReAct 工作模式，为 AI 提供可以异步调用的多种工具，构建自主任务执行循环的能力。在任何任务执行中，都应该帮助 AI 提供自主判断任务执行结果的验证方法，此方法需要异步调用并及时返回验证结果（以下观察阶段）。这有助于 AI 形成自闭环，提升任务执行效率和质量。

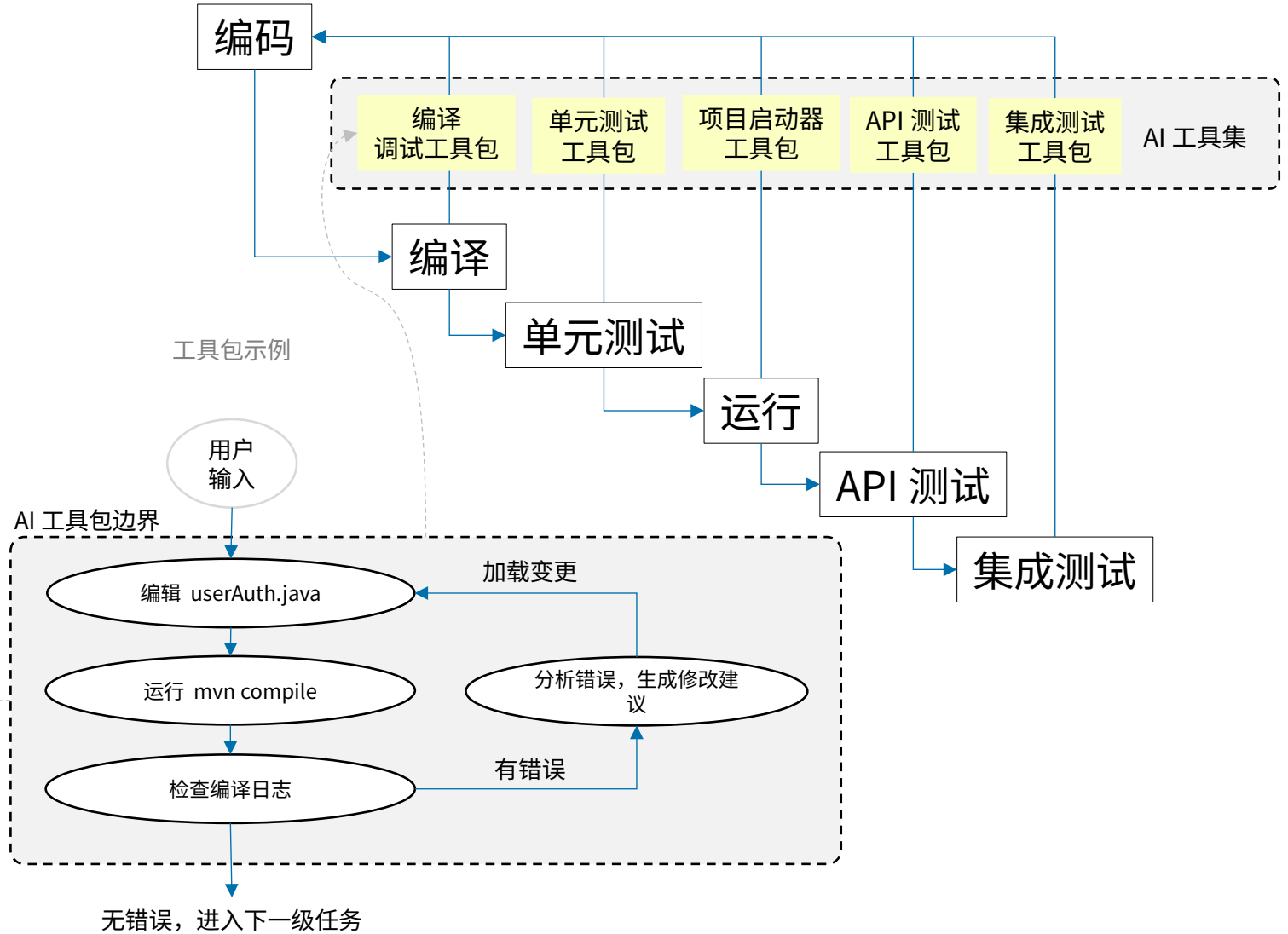


工具引擎

对接方式： MCP/SKILL

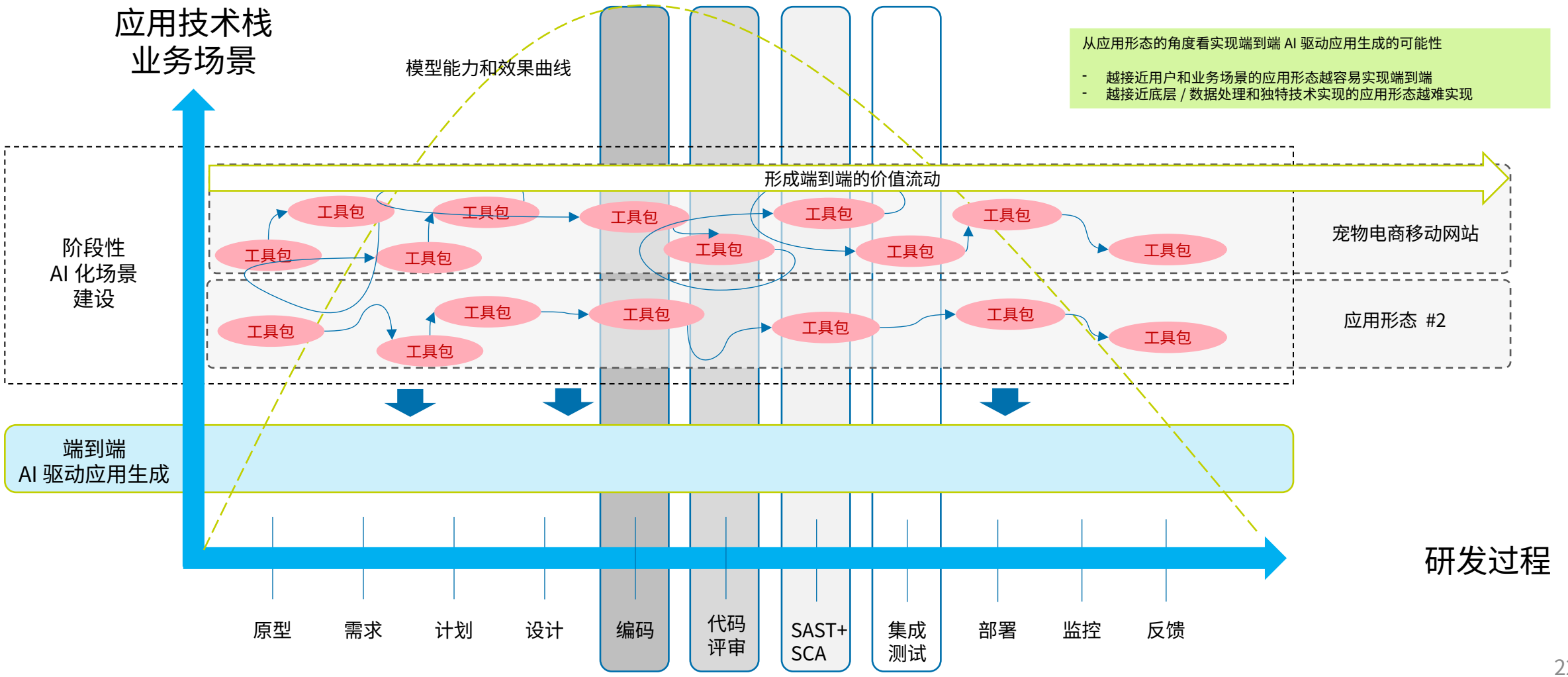
复用

AI Native 工程平台



AI4SE 整体实施策略 - 一横四纵 构建研发智能体集群

- 横向布局：在研发各环节逐步构建或者引入 **ASDM 工具包** 提供模型对特性研发环节的认知和执行能力，逐步串联成端到端能力。
- 纵向布局：那些与应用形态关联性不强的通用领域，比如：编码、代码评审、代码检查和修复、安全性检查和修复；比较容易形成能力积累，产业界的投入也最聚焦；会最早形成较为完整的 AI 驱动提效工具链。其他环节由于业务属性更强，不同企业具备更大的差异性，对应用形态比较独特的企业更依赖自身能力进行构建。



AI4SE 落地运营建议 – 构建开发者心智，提供 AI 化场景 能力

- 核心实践 1 - 推动开发者从 **代码补全** 向 **Agent 编码** 模式切换
- 核心实践 2 - 推动 **最佳实践集** 和 **AI 工具包 /AI 化场景** 的持续建设

全面构建 AI Native 工程平台 规划方案

交付物 1
试点团队掌握并落地核心实践

1. 建立项目级认知并在项目中验证效果
2. 构建 AI 工具包并在项目中验证效果
3. 赋能试点团队成员，掌握核心实践技能，提升项目团队效能

试点团队收益

1. 在日常项目中可用的项目级认知和 AI 工具包，可见的效能提升（指标）
2. 总结实践经验，输出到文档中心；组织内部分享，推动更多团队采用

交付物 3
内部推广机制

1. 通过持续全员培训，分享推动更多团队加入 最佳实践文档中心 的使用和更新
2. 试点团队作为种子团队，利用示范效应拉动更多团队加入 最佳实践文档中心的建设

交付物 4
AI Native 工程平台设计方案

1. 协助内部 DevOps 平台在设计中引入 AI Native 设计思路
2. 为整个招商证券引入 AI4SE 能力提供标准化支撑，降低使用门槛，提升整体效能

交付物 2 – AI4SE 最佳实践文档中心

- ✓ 如何构建项目级认知和相关模版，提示词，操作指引
- ✓ 如何构建 AI 工具包和相关开箱即用工具包
- ✓ 培训教材，通用实践操作手册
- ✓ 其他试点项目总结

AI4SE 建设效果评价体系 – 5 层评价、逐层推进、鼓励创新

以下评价维度代表本项目的交付目标，围绕这些目标明确价值并使用指标证明价值的达成和提升。措施部分则对应到具体的实施工作内容，在本方案的后续部分进行说明。

评价维度	价值体现	指标	措施
平台基础	平台稳定可靠 功能可用好用	运维相关指标（系统性能指标、并发 / 响应时长） 服务端 / 插件端升级成功率 问题反馈数量 / 关闭速度	• 持续关注平台稳定性指标 • 持续关注服务端 / 插件端升级成功率 • 持续关注并推进问题闭环
实践应用	掌握并熟练掌握各项功能 最大化发挥各项功能的提效能力	用户活跃度（总用户数、总活跃人数，活跃率） 补全（采纳率、生成率） 对话（次数） Agent （次数、代码生成量）	• 产品基本功能性培训 • 场景化实施和实操指导 • 集成场景探索、总结、设计和实现 • 相关指标解读、采集和趋势统计
场景质效	场景产能提升 场景覆盖 / 保持稳定	场景效率提升率、累计效能收益、场景覆盖数量、场景落地成功率	• 关指标解读、采集和趋势统计
交付质效	需求交付吞吐量提升 质量提升 / 保持稳定	变更时效、变更月均交付量，缺陷密度	• 相关指标解读、采集和趋势统计
价值创新	AI 原生最佳实践总结 AI 原生工具构造和推广	最佳实践条目数、分享次数，参与分享人数、点赞数 AI 原生工具的产出数量和使用情况	• 指导并推动团队总结和分享最佳实践 • 指导并推动团队开发 AI 原生工具并在内部推广

AI4SE 建设效果评价体系 – 思路转变

- 忘记 **代码补全**，转向关注 **Agentic Coding 场景化提效**
- 忘记 **补全采纳率、补全生成率**；转向 **AI 化场景 数量和 单场景提效比例**

关注过程 → **关注结果 / 产出**

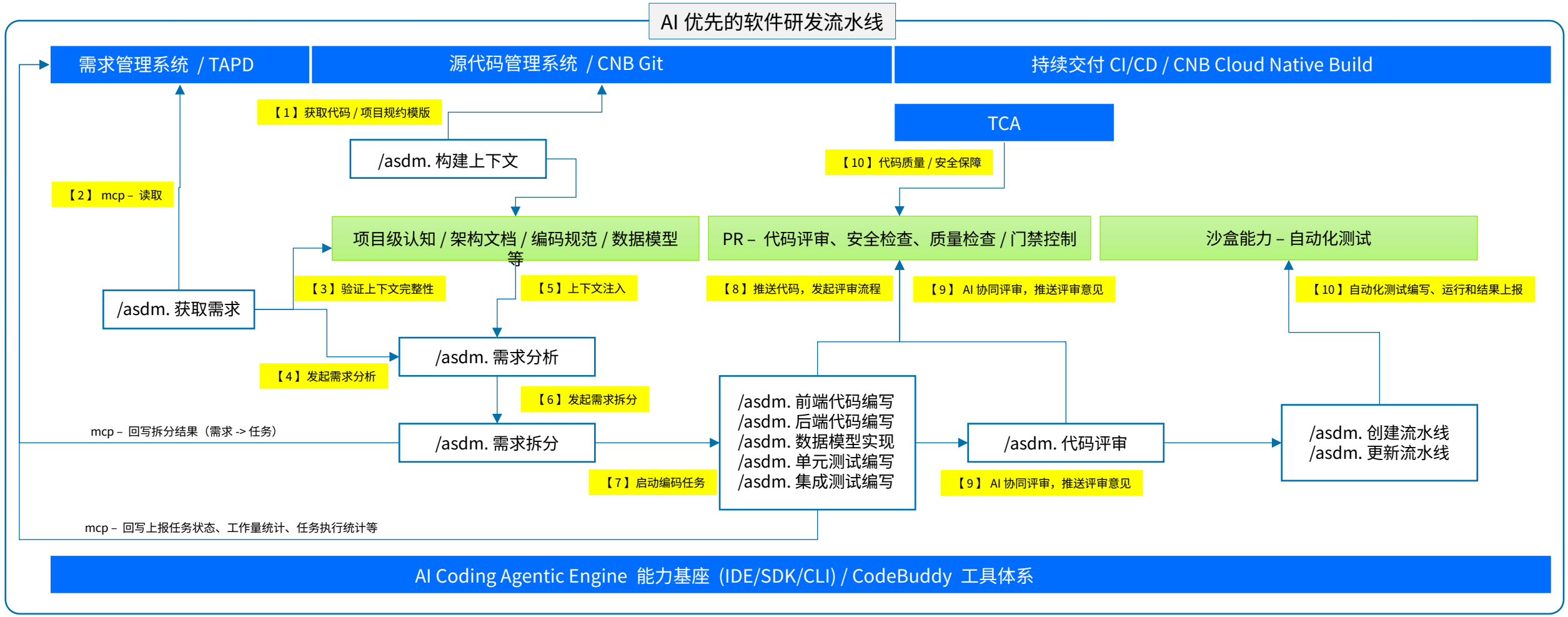
PART 03

ASDM 管理平台
AI 驱动开发工作流的运营中枢

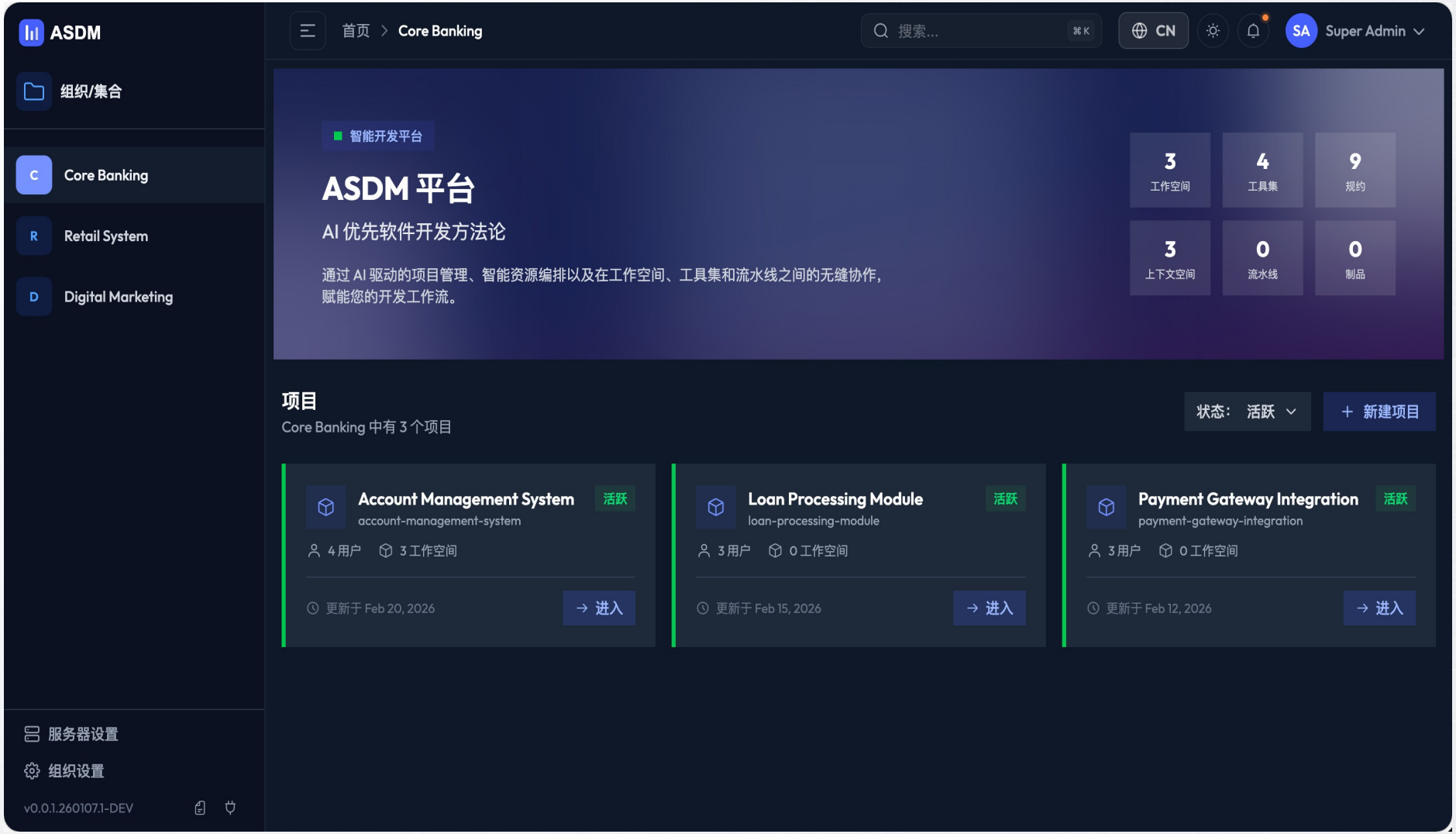
AI 优先的 AI4SE 软件研发流水线 – 全景规划

基于 ASDM 的 AI 优先方法论，我们通过持续构建原子能力（ASDM 工具包），并将其深度集成于关键研发环境，将传统研发中依赖人工的工艺环节，系统性地转变为 “AI 主导、人工辅助” 的工作模式。这使得 **AI 成为软件研发流程中的核心价值创造者**。人的角色因此从直接产出软件，演进为专注于设计、构建与持续优化 AI 驱动的研发流水线。借助将企业研发策略通过各类**规约进行显性化定义与版本化管理**，这一转变不仅能为企业带来端到端的 AI 价值输出，更能**实现传统模式难以企及的研发过程标准化、规范化，以及卓越的质量与安全保障。**

需求与设计 → 架构与规划 → 编码和测试 → QA 与安全 → DevOps 与持续交付



ASDM 平台 - AI 驱动开发工作流的运营中枢



集中化管理
跨项目统一管理所有 ASDM 资源的界面。

广泛兼容性
支持多种 AI 开发工具，包括 OpenClaw, Claude Code GitHub Copilot Cursor Cline CodeBuddy 等。

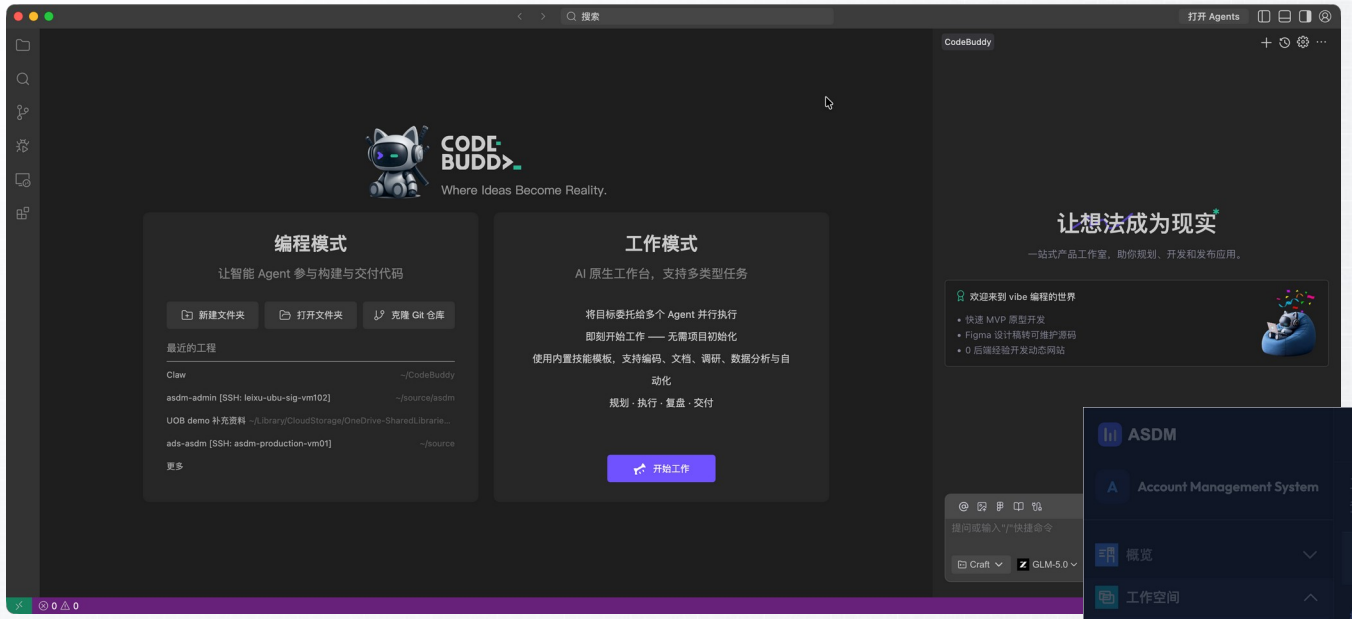
团队协作
支持多用户和企业级角色的访问控制。

可扩展架构
基于插件的系统，支持集成自定义工具和工作流。

企业就绪
为生产环境构建安全性、可扩展性和可靠性。

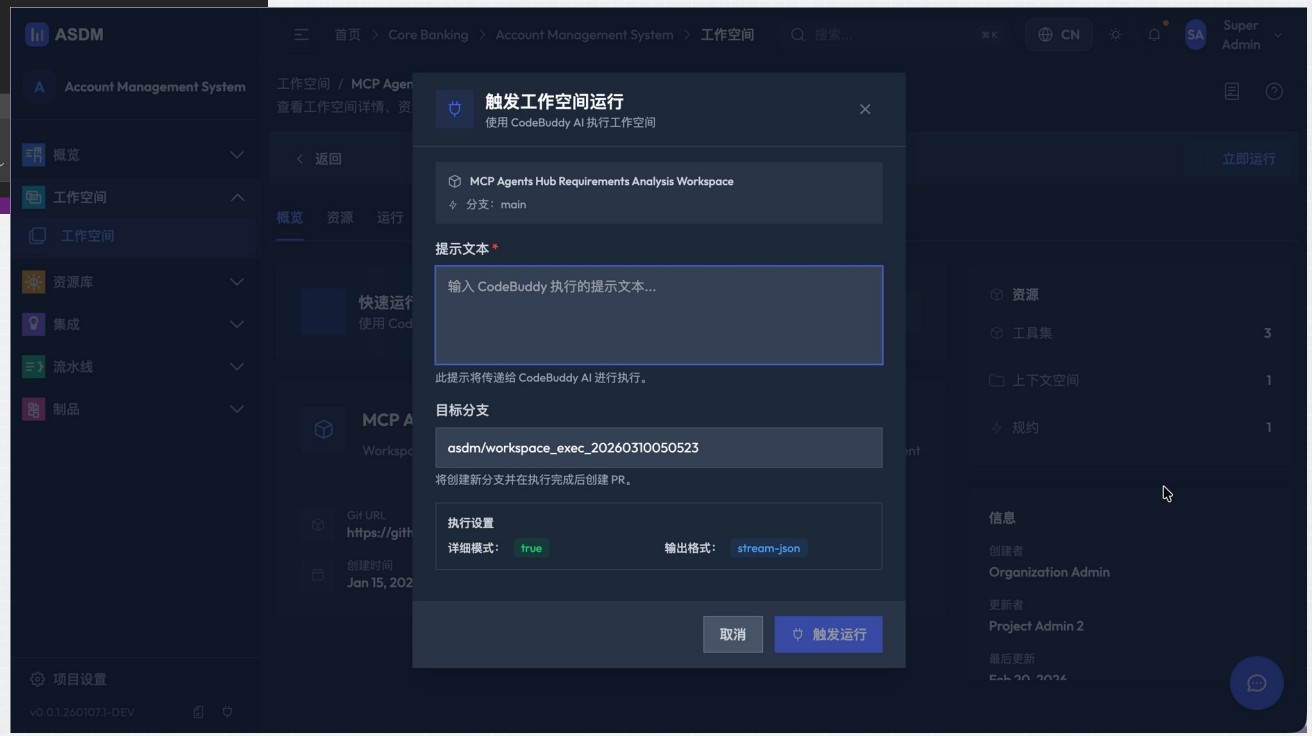
AI 优化设计
从底层构建，专为支持 AI 驱动开发工作流而设计。

ASDM 平台 – 双模工作升级 // CodeBuddy Claw 集成



CodeBuddy 0309 Update

- 支持编程 / 工作两种工作模式
- 支持本地云端双模工作区



ASDM 0309 Update

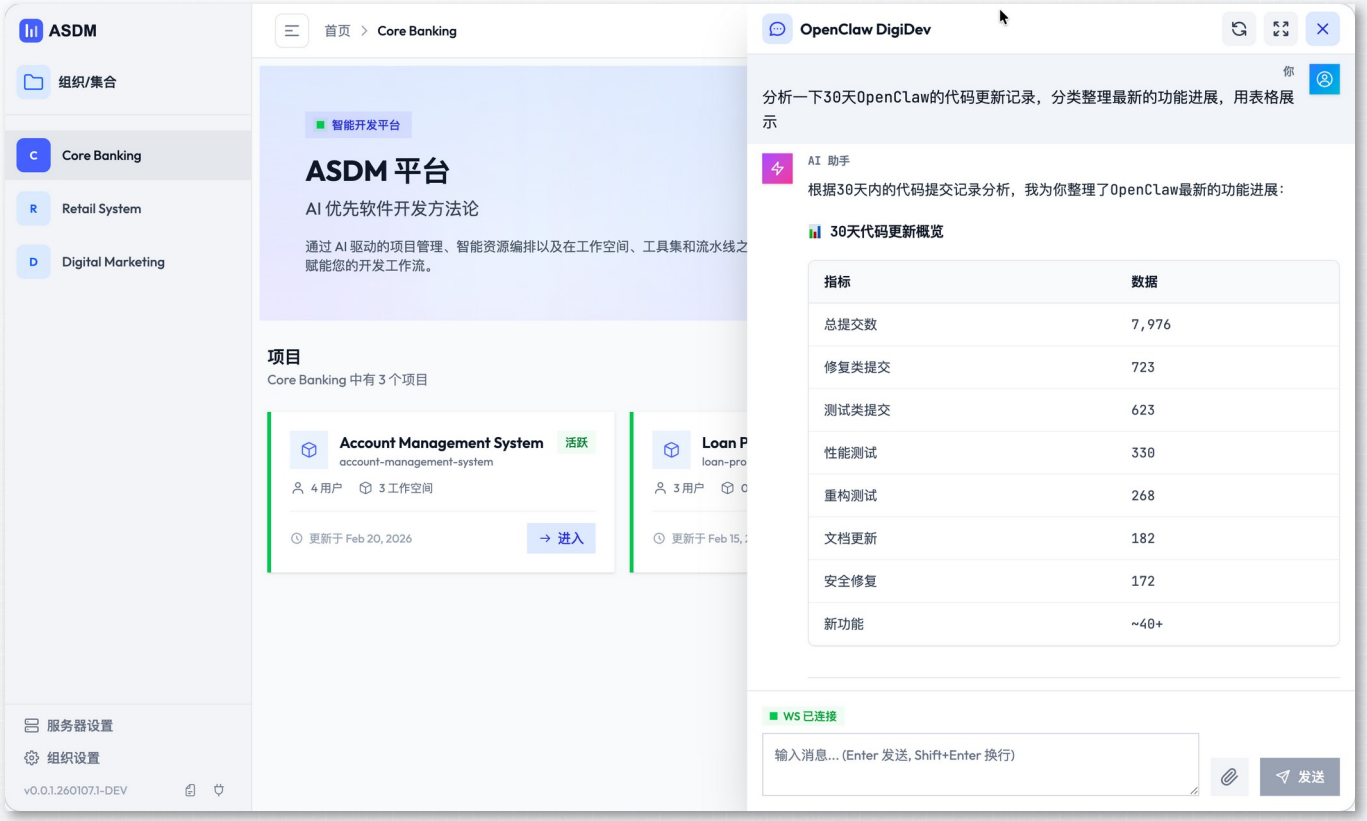
- 支持 DigiWorker 数字分身
- 支持在线 Workspace 任务执行

ASDM 平台 DigiWorker – 工程师的数字分身 // 懂你所想，知你所需 //

隔离的 OpenClaw 执行环境 – 确保安全执行

为不同开发场景创建专用工作空间。每个工作空间为 AI 代理提供隔离的执行环境，支持可配置的上下文和工具集。

- 配置工作空间特定的上下文和工具集
- 跟踪工作空间历史和变更
- 与 AI 代理工作流无缝集成

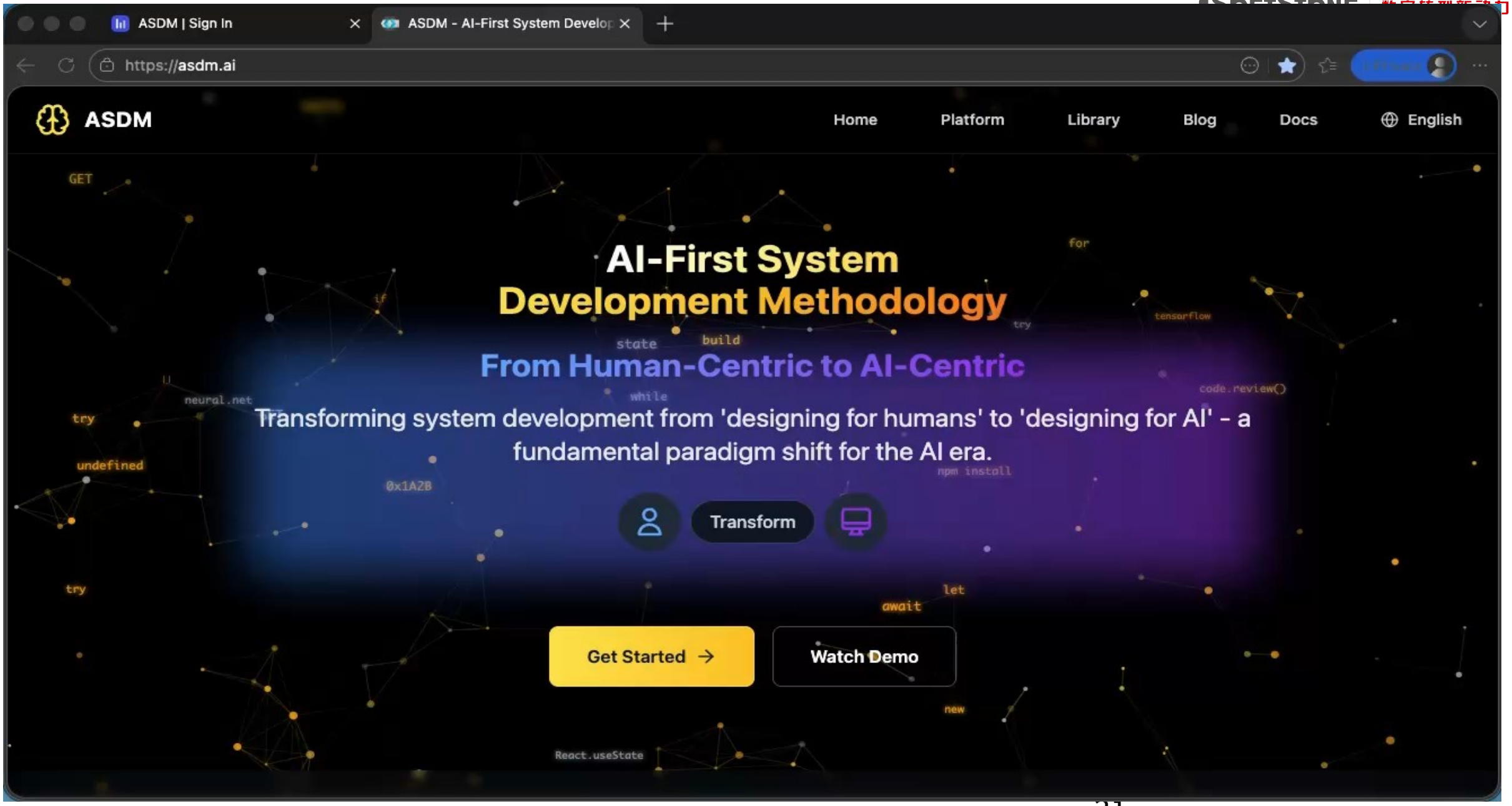




为 OpenClaw 提供有效的上下文和稳定执行能力

为 OpenClaw 提供理解企业研发环境的上下文和专业化编码工具集。

- 为企业提供开箱即用的养虾环境
- 为 OpenClaw 提供对企业环境的准确感知和检索能力
- 释放企业对智能体调度的丰富诉求


ASDM | Sign In
ASDM - AI-First System Develop
https://asdm.ai
ASDM
Home
Platform
Library
Blog
Docs
English
AI-First System
Development Methodology
From Human-Centric to AI-Centric
Transforming system development from 'designing for humans' to 'designing for AI' - a fundamental paradigm shift for the AI era.
Transform
Get Started →
Watch Demo

AI4SE 软件研发流水线：支撑体系架构

工具支持

CodeBuddy IDE – AI 原生 IDE

CodeBuddy Plugin – IDE 插件

CodeBuddy CLI – 基于命令行的 AI 变成解决方案

知识库 /Rules | 成员管理 | 研效看板 | 智能体管理 | 日志管理 | 模型 / 网络

Admin 统一管理后台

Deepseek | 混元系列 | 私有模型 | 自定义模型

企业实践

后台管理

需求管理 | 文档管理 | 权限管理 | 日志管理

规范模板（项目、需求、开发、测试）

度量（插件、用户、质量、成本）

MCP Servers

后台管理 MCP Server

MasterGO MCP Server | Test MCP Server

Sec MCP Server | DevOps MCP Server

SKILLS & Tools

Git 操作 | 扫描执行分析

自动化测试 | 报告生成

理论基础

构建项目级认知

技术问题调研和解答

逆向工程 / 文档维护 / 管理系统数据维护

需求、方案、概设 | 数据、接口、详设、 WBS | 代码补全、规范保障、测试 | 规范、安全、评审、度量 | CI/CD 、运维、反馈

AI4SE 软件研发流水线：后台管理 – 企业级规约 (spec) 集中管理能力

为了支撑 ASDM 体系在企业内的可管理性，提供后台管理门户并包含：规约模版管理、过程跟踪管理、交付物管理、可观测性四个模块用于支撑在多项目多团队协同研发场景下的标准化和差异性需求。通过将 ASDM 工具包中的模板在后台统一管理，实现企业级管理策略的统一下发和及时更新；通过唯一的需求 ID 形成端到端的跟踪；通过对产出物的统一管理形成数据沉淀，以便支撑 AI 能力的自我迭代和改进。

管理后台包含 规约模版管理、过程跟踪管理、交付物管理、可观测性 四个核心功能模块，各模块功能及流转逻辑如下：

规约模版管理模块

- 提供 至少 7 类通用的规范模板：包含代码规范模板、 UI 规范模板、需求规范模板、安全规范模板、测试报告模版、概要设计模版、项目模版，作为全流程的约束基准；需求模版需要包含平台地址、可测试账密、 API 文档地址，方便生成合理的测试计划。
- 支持自定义模板：可通过 “代码 / 原型示例 + 自然语言” 的方式，自主定义符合业务场景的专属规范模板。

过程跟踪模块

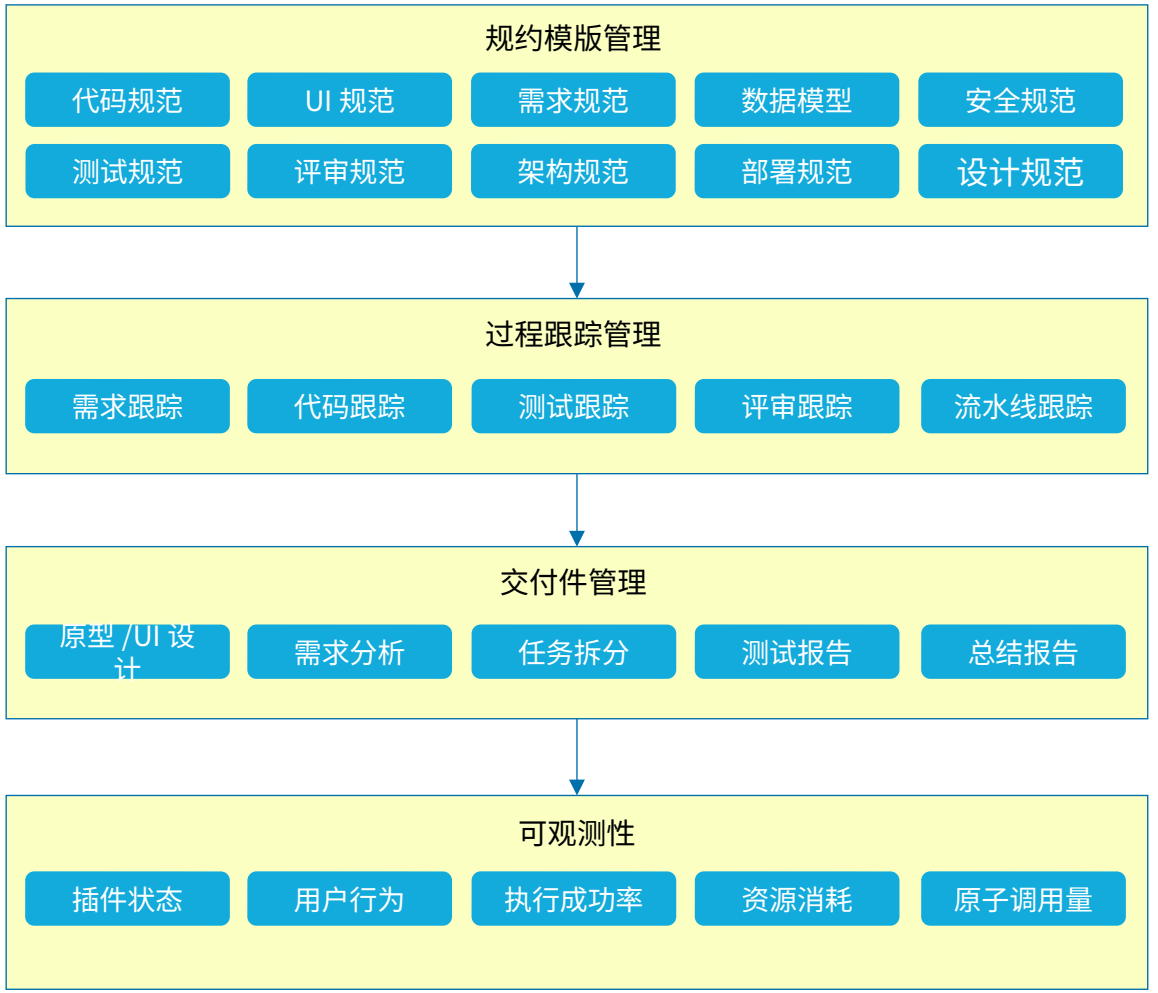
- 需求创建：自动生成唯一需求 ID，作为需求全流程的标识。
- 需求关联：支持上传 / 绑定外部需求文档，并关联模板管理中的规范模板（代码 / UI / 需求 / 安全类）。
- 状态跟踪：实时展示需求对应的报告关联状态（“待关联报告” “已完成”），实现需求与后续输出报告的精准绑定，便于全链路追溯管理。

交付件模块

- 承接需求处理流程，统一保存管理全流程产出的各类成果物，支持在线查看：
- 输出物包含：原型 / UI 设计稿、概要设计文档、联调报告、测试报告、安全报告、总结报告。
- 作用：集中存储、展示全流程的核心交付文档，便于查看与追溯。

可观测性模块

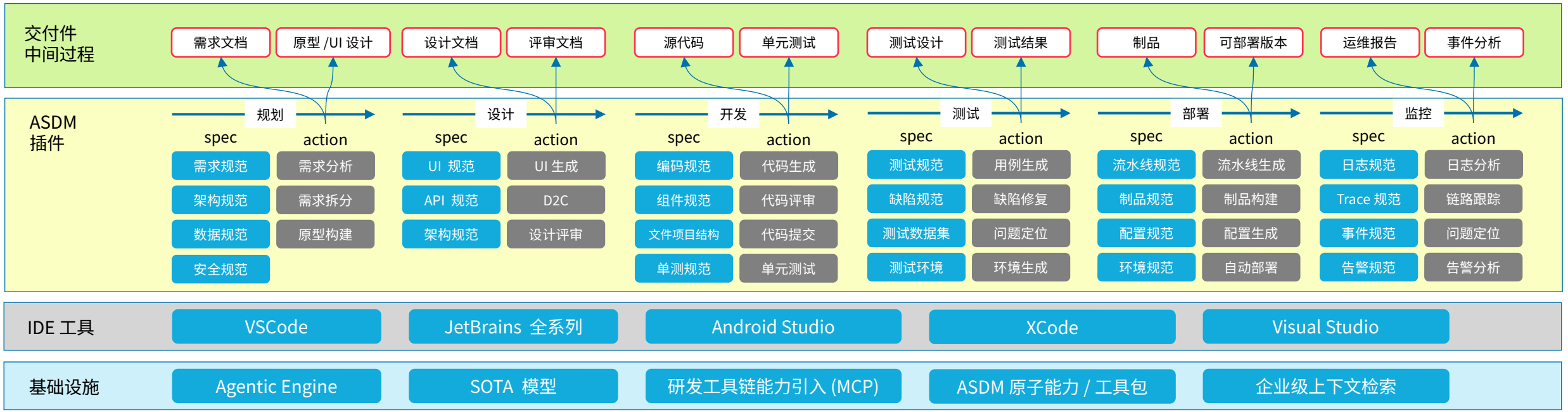
- 对插件全链路实时监控、数据沉淀与价值量化，支撑插件稳定运行、体验优化与价值验证。
- 插件状态：监控插件兼容性（按 IDE / 系统统计在线、启动成功率），捕获崩溃 / 功能失效等异常并记录，统计 IDE 版本分布情况。
- 用户行为：用户活跃度统计、功能使用频次统计、需求场景分析、用户操作漏斗分析等。
- 执行质量：量化指标总览、效果趋势分析、失败案例复盘等。
- 效率成本：流程耗时统计、资源消耗成本统计、呈现落地价值评估等。



AI4SE 软件研发流水线： ASDM IDE 插件 – 提升操作规范性、提高推广效率

SOFTSTONE 软通动力 | 数字转型新动力

通过提供 IDE 插件，为开发者提供简单易用的研发流程操作界面，使用 **ASDM 原子能力 / 工具包** 闭环覆盖 **规划→设计→开发→测试→部署→监控** 全流程，需实现各环节的自动化 / 半自动化联动，着重确保在成本可控的目标下，实现测试与修复的多轮执行，最终生成生产级别的高品质代码。同时减少开发人员学习成本，提高推广效率。



项目上下文理解阶段

- 代码拉取：支持从指定代码仓库（如 Git ）拉取存量代码的能力。
- 代码分析：自动解析存量代码的技术栈、组件、规范等信息。
- 代码总结：生成存量代码的结构化总结报告，作为后续设计 / 开发环节的参考依据。

设计阶段

- 原型 / UI 设计：支持通过自然语言生成 UI 设计稿，并具备设计稿转 HTML 代码的能力。
- 概要设计：自动生成概要设计文档，同时输出单元测试用例、功能测试用例。

开发阶段

- 自动编码：基于技术约束、规范、需求等前置信息，实现前端代码的自动生成。
- 自动调试：提供代码自动调试能力，支持调试结果的 “反思修复” 闭环。
- 安全管理：支持开发分支的自动创建、代码安全扫描、扫描报告生成，以及扫描问题的反思修复。

测试阶段

- 测试用例生成：自动生成匹配需求的测试用例。
- 测试执行辅助：提供测试执行的辅助工具 / 能力。
- 问题分析：支持测试过程中问题的自动分析与记录。

部署阶段

- 发布应用：支持生成代码的自动化部署与应用发布能力。

AI4SE 软件研发流水线：中间层产物管理

中间层产出物需要跟管理后台的需求 id 进行关联，并把文档存储在管理后台，方便回溯查看。对多轮循环产生的修改记录，需要关联到文档的修改记录中，方便回溯查看。文档需在必要时候做截图 / 录屏 / 日志记录，做好分类（如标记缺陷等级），关联需求 ID 和代码路径。

中间产物名称	产出对应环节	详细需求描述
原型 / UI 设计稿	设计阶段 - 生成 UI 设计稿环节	按照需求文档以及 UI 规范生成。
概要设计文档	设计阶段 - 前端概要设计环节	包含前端架构、组件规划、技术选型等核心内容，作为开发环节的技术依据。
单元测试用例文档	设计阶段 - 单元测试用例环节	匹配前端代码的单元测试场景，覆盖组件、工具函数等最小测试单元。
功能测试用例文档	设计阶段 - 功能测试用例环节	完整覆盖业务需求对应的功能验证场景，包含输入、预期输出、执行步骤等信息。若涉及不同用户角色的使用，通过预先准备好不同身份的登录信息进行测试；若存在多语言情况，自动切换语言进行测试。能够覆盖微小图标、字体等细节。自动启动浏览器执行测试，无需人工操作，支持 PC / 移动端多环境测试。
源代码	开发阶段 - 自动编码环节	能够依据项目所设定的技术栈（如 React 、 Vue 等）以及前端通用规范（包含代码格式、编译规范、样式规范），自动生成符合该技术栈规范和最佳实践的代码。生成代码需自带注释，代码无高危安全漏洞。仅对本次需求涉及的范围进行改动。若遇到非本次需求范畴，但代码存在不合理之处或有 bug 的情况，需由人工决定是否进行修改。
联调报告	开发阶段 - 调试环境	依据 API 文档生成 mock 数据，并进行前后端功能联调。如非阻塞点能自动完成代码修复；如遇到阻塞点（如：后端 API 异常 或者 其他系统联动）能够在界面上反馈结果并支持人工介入，处理完成后自动开展代码修复。
安全扫描报告	开发阶段 - 代码扫描环节	包含安全漏洞类型、风险等级、影响范围、修复建议等核心内容。
测试报告	测试阶段 - 测试执行环节	覆盖测试用例执行结果、问题数量统计、问题严重程度分布等信息。如非阻塞点能自动完成代码修复；如遇到阻塞点（如：后端 API 异常 或者 其他系统联动）能够在界面上反馈结果并支持人工介入，处理完成后自动开展代码修复。
总结性报告	全流程收尾汇总环节	整合所有中间产物信息，输出内容包括但不限于：流程效率类指标：全流程总耗时，以及各个环节的细分耗时。全流程总 token 消耗，以及各个环节的细分 token 消耗。设计→编码→测试环节的总迭代轮次、单次迭代平均耗时。质量效果类指标：阻塞点中断次数以及详情。涉及代码范围总结，功能影响面枚举。缺陷修复闭环率（缺陷从发现到修复的完成比例）。需求缺失项初始数量、补充完善的需求项数量、需求变更次数。说明需求整体改动点。合规与规范类指标：安全报告中漏洞修复完成率、合规检查项（如等保要求）的达标比例。

AI4SE 软件研发流水线： ASDM 工具包生命周期 / 跨项目上下文索引

SoftStone 软通动力 | 数字转型新动力

将 ASDM 中的实体对象进行组织，包括： Agentic 工作引擎，工作空间（workspace），工具包，工具包仓库，工具包分发工具，上下文注册表组织成一个系统。形成从工具分发，模版化能力管理，企业级上下文引入到 Agentic Engine 自反馈循环能力构建的 原子能力 生态闭环。

工具包分发

1. 安装 ASDM 分发工具 或 IDE 插件

ASDM 将提供一个分发工具，此工具可能通过 IDE 插件的形式提供。开发人员通过 VSCode 或者 JetBrains 插件市场安装此工具到自己的 IDE 环境。

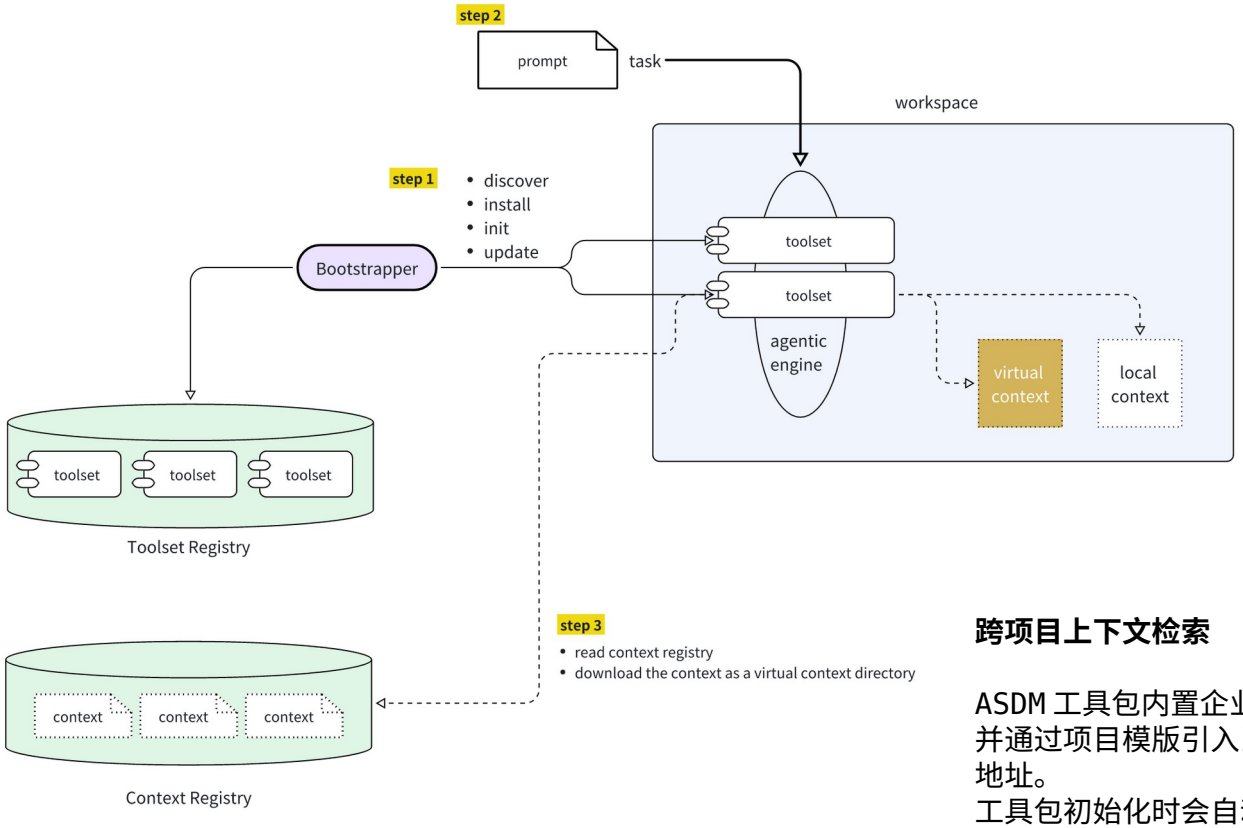
2. 选择和发现工具包

开发人员通过 ASDM 分发工具插件，浏览，查看或者搜索 ASDM 工具包仓库，选择需要的工具包。

3. 安装工具包

开发人员点击工具包列表中对应的安装按钮， ASDM 分发工具开始下载并将工具包放置于开发人员本地工作目录，一般为当前项目根中的 .asdm 目录。

工具包一般为一个 zip 压缩包文件，下载完成后会被解压缩到 .asdm/toolsets 目录下。



跨项目上下文检索

ASDM 工具包内置企业上下文注册表地址，并通过项目模版引入当前项目的注册表示例地址。

工具包初始化时会自动下载注册表索引文件到本地 .asdm/context 目录，于其他本地上下文索引同属相同的文件目录。

Agentic Engine 读取跨项目上下文索引后，根据需要按照索引文件说明，调用 mcp-server 进行跨项目上下文检索。

AI4SE 软件研发流水线：场景能力 /ASDM 工具包（Toolset)/ 原子能力 / 智能体

SOFTSTONE 软通动力 | 数字转型新动力

工具包（Toolset）是 ASDM 的交付单元，每个工具包都是一个可以**独立开发、测试、打包和持续交付**。同时，工具包被设计成独立于**工作空间（workspace）**的模式，可以在不同形态的 Agentic Engine 中提供统一的工作编排和执行效果，包括： IDE 插件，独立 IDE， CLI 或者 Background Agent 。通过维护统一的 工具包 (Toolset)，企业可以形成标准化的 AI 能力原子，并在不同的环境中提供对这些原子的执行能力。

工具包可以通过**斜杠指令**或者**定制化智能体**的方式供用户进行快捷调用，首批 ASDM 工具包 集合包括以下原子能力。

领域分类	场景	原子能力 斜杠指令 / 智能体	特性	价值
核心能力	上下文管理	/asdm. 构建上下文	用户通过 CodeBuddy 斜杠指令自动完成项目级（单个调试单元）的 AI 上下文构建，此上下文通过 **.asdm/context** 目录进行存储，用来辅助 AI 对项目的业务背景，架构，数据模型， API ，编码规范等内容进行了解，提升 AI 在执行后续任务过程中对项目个性化信息的理解能力	解决 AI 对项目个性化信息的认知问题，业界提供了各种工具解决这个问题，但是都需要开发者本身具备一定的技能。这个特性的设计目标是让这个过程自动化，无需开发者学习这些技能也能让 AI 充分 / 准确的理解当前项目。
需求	需求分析和拆分	/asdm. 获取需求 /asdm. 需求分析 /asdm. 需求拆分	用户可以使用 CodeBuddy 斜杠指令自动从需求管理系统（ TAPD ）中获取需求描述，并由 AI 完成需求分析和拆解，形成可以直接用来进行编码的任务描述，此任务描述面向 AI 提供，为后续 AI 自动根据此描述完成编码提供输入。	解决需求分析和拆分过程必须要需求人员和资深开发者协作完成的问题，需求分析需要根据项目业务背景，技术特点完成，这些信息的来源原依赖需求人员和资深开发者的经验。此特性允许 AI 替代人工完成这个分析过程，由 CodeBuddy 自动基于 **/asdm. 构建上下文** 中的上下文索引完成对代码库和相关心的综合分析，并给出符合要求的分析结果和拆分任务。
编码	编码开发	/asdm. 前端代码编写 /asdm. 后端代码编写 /asdm. 数据模型实现 /asdm. 单元测试编写 /asdm. 集成测试编写	用户可以使用 CodeBuddy 斜杠指令自动完成所支持技术栈的编码任务，包括：前端代码编写（ React.js ），后端代码编写（ Java SpringBoot ），数据模型设计和数据结构迁移 / 升级（ PostgreSQL/Redis ），前端单元测试编写（ React.js+Mocha.js ）并满足目标代码覆盖率要求，后端单元测试编写（ Java ）并满足目标代码覆盖率要求	此特性允许 AI 替代人工自动完成经过 **/asdm. 需求分析和拆分** 出的任务的编码过程，并确保所编写代码满足业务需求和基本的质量要求。
评审	代码评审	/asdm. 代码评审	用户可以使用 CodeBuddy 的斜杠指令或者 CNB PR 自动触发 AI 代码评审，并且允许项目管理员通过配置评审规则对评审的侧重点进行配置。CodeBuddy 斜杠指令评审发生在 IDE 中，一般针对当前待提交代码进行； CNB PR 的 AI 代码评审发生在 PR 过程中，一般针对当前特性分支中完整进行评审	此特性允许 AI 替代人工完成对待合并代码的评审能力，结合预设在模版中的规则对待提交代码完成。此特性的设计目标是允许项目管理员根据项目特性设置当前项目的评审关注点，确保所提交代码符合项目编码规范。此特性使用 **/asdm. 构建上下文** 中所产出的相关项目编码规范作为输入模版。
持续交付	流水线构建	/asdm. 创建流水线 /asdm. 更新流水线	用户可以使用 CodeBuddy 斜杠指令自动完成适合当前调试单元代码特性的 Jenkinsfile 的编写，同时此 Jenkinsfile 也满足通过管理员所定义的企业级流水线规范。提交代码后流水线自动生成并触发。此流水线根据企业管理员所预设的规则完成测试环境和生产两套环境的部署。	此特性允许 AI 替代人工完成流水线配置文件的编写，并结合预设在模版中的规则生成复合企业流水线规范和基础设施配置的可以执行流水线。此特性同时提供针对流水线报错进行对应修复的能力。为团队提供全智能化的流水线维护能力。

PART 04 AI Coding 超级个体培训

培训背景

AI Coding 是当前生成式 AI 领域最具确定性的生产力场景，随着模型和 AI Coding 工具的日渐成熟，已经有大量开发者成为了 超级个体开发者

- 他们可以使用简单精炼的提示词准确的控制模型的行为；
- 他们可以在几小时内完成普通开发者几周的工作量；
- 他们可以快速掌握新的开发语言 / 框架 / 技术，他们也可以在不同的领域穿梭自由，无论是前端还是后端、无论法律还是医学，无论金融还是投资；他们都可以快速掌握其中的知识精髓，构建出高价值软件产品 / 项目，并赚到钱。

更重要的是，他们可以利用先进模型和 AI Coding 工具为自己和团队创造前所未有的价值，在 AI 时代获得生存空间。同时，你是否意识到，如果你不能掌握这些技能，你就是那个被迅速淘汰的 旧时代开发者。超级个体开发者 可以以高于你几百倍的速度完成工作，构建软件，创造价值；旧时代的开发者自然无处可逃，面临被无情淘汰的命运。

成为超级个体开发者，不是一个选项，而是你职业生涯的一次选择，留下还是离开，如此简单，也如此残酷！！



项目收益

五大核心收益



收益 1：

了解生成式 AI 代码生成的
工作原理，从而充分掌
握引导模型的底层思维逻辑



收益 2：

熟练掌握主流 AI Coding
工具的各项功能，包括：
腾讯 CodeBuddy，
阿里 Tongyi/Qoder，
字节 Trae， GitHub Copilot，
Claude Code， Cursor 等



收益 3：

熟练掌握软件编码领
域的提示词工程和上
下文工程技巧，从而
具备将 AI 模型训练成懂项
目的专业模型的能力



收益 4：

熟练掌握各类业界先进
工具和实践并灵活运用
包括： Rules, Subagent,
MCP， Skills 和 SSD；
从而具备在企业级项目
中应对各类复杂场景的能力

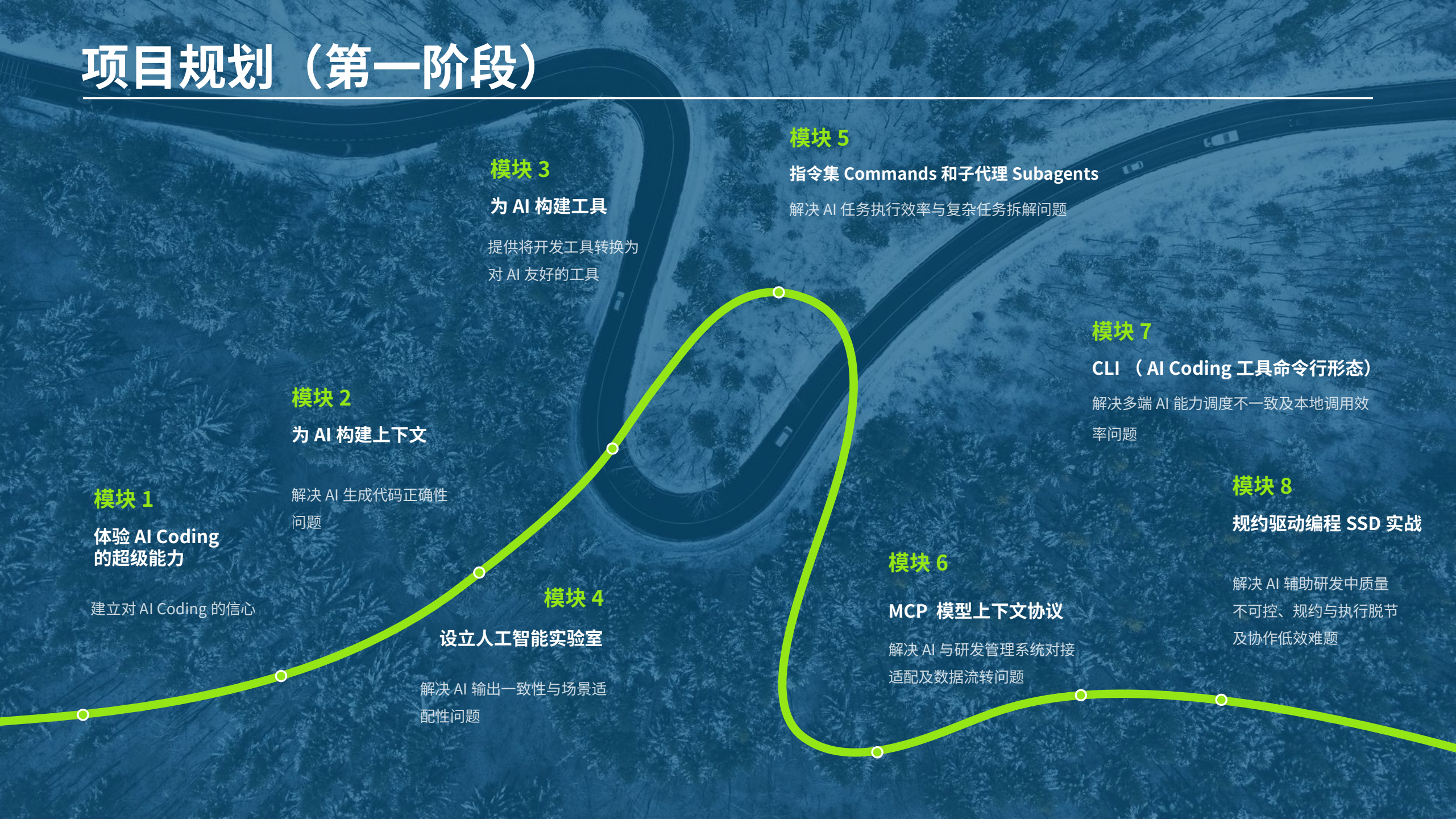


收益 5：

具备成为 AI Coding
架构师的基本技能，
从而具备辅助团队
完成 AI4SE 转型的能力

“旧时代开发者” —— **快速跃迁** ——→ **“AI 时代的超级个体开发者”**

项目规划（第一阶段）



模块 1

体验 AI Coding 的超级能力

建立对 AI Coding 的信心

模块 2

为 AI 构建上下文

解决 AI 生成代码正确性问题

模块 3

为 AI 构建工具

提供将开发工具转换为对 AI 友好的工具

模块 4

设立人工智能实验室

解决 AI 输出一致性与场景适配性问题

模块 5

指令集 Commands 和子代理 Subagents

解决 AI 任务执行效率与复杂任务拆解问题

模块 6

MCP 模型上下文协议

解决 AI 与研发管理系统对接适配及数据流转问题

模块 7

CLI （AI Coding 工具命令行形态）

解决多端 AI 能力调度不一致及本地调用效率问题

模块 8

规约驱动编程 SSD 实战

解决 AI 辅助研发中质量不可控、规约与执行脱节及协作低效难题

课程详情

模块	内容	目标	形式	考核形式
模块 1	**体验 AI Coding 的超级能力** 跟随讲师一起完成一款企业级应用的从 0 到 1 构建，实际体验完全使用提示词，不写一行代码，在 4 小时内构建一款企业级应用，包括： Vue.js 前端， JavaSpringboot 后端服务， MySQL 数据库和容器化环境	**建立对 AI Coding 的信心** 通过这 4 个小时的演练引导模型完成这个应用的从 0 到 1 ，相信你会充分信任 AI Coding 是可以胜任企业级软件开发的各类场景的。这个过程中，你也会了解到 GenAI 生成代码的工作原理 以及 四段式提示词的设计精要 两个核心技能，这两项技能将成为你成为一名 超级个体开发者 的能力基础。	4 小时演示讲解 4 小时动手实验	测评 + 实操作业
模块 2	**为 AI 构建上下文** AI 模型的认知问题直接影响生成代码的准确性和方向性。本模块提供清晰的上下文构建步骤和常用的上下文构建技巧，让你的 AI 模型迅速成为懂你的朋友和懂项目的专家	**解决 AI 对企业项目的认知问题** 让 AI 有效的理解你的项目的核心不是建立知识库，而是建立高效的检索机制。本段课程中，你将了解到 Agentic Coding 引擎的检索机制 以及 上下文构建 5 步工作法 ，让你的规则集 Rules 真正有效起来。这是我们为 AI 构建软件工程系统 的第一步，也是你成为 超级个体开发者 的关键步骤。	2 小时演示讲解 2 小时动手实验	测评 + 实操作业
模块 3	**为 AI 构建工具** 本模块提供将开发工具转换为对 AI 友好的工具，方便 AI 进行调用，并利用 agent 闭环工作方式为 AI 提供及时准确的反馈，最终实现改进 AI 生成效果的目标。	**解决 AI 生成代码正确性问题** 生成代码的正确性保障不能仅仅依赖模型和提示词，而要通过建立 PDCA 反馈环的方式来实现。在本段课程中，我们将介绍如何将常用的开发工具构建成适合 AI 调用的模式，从而解决代码生成准确性问题。	2 小时演示讲解 2 小时动手实验	测评 + 实操作业
模块 4	**规则集 Rules 和技能包 Skills** 本模块聚焦 AI 应用的核心底层支撑，通过构建标准化规则集与场景化技能包，规范 AI 行为边界、强化 AI 任务处	**解决 AI 输出一致性与场景适配性问题** AI 在复杂业务场景中易出现输出偏差、逻辑混乱、场景适配不足等问题，仅靠工具调用难以从根源规避。本段课程将详解规则集的制定逻辑与技能	2 小时演示讲解 2 小时动手实验	测评 + 实操作业

课程详情

模块	内容	目标	形式	考核形式
模块 5	**指令集 Commands 和子代理** Subagents 本模块聚焦 AI 任务执行的高效调度体系，通过设计标准化指令集与模块化子代理，实现任务的精准拆解、分工协作与高效落地，衔接前文工具、规则与技能，构建完整 AI 工作流闭环。	**解决 AI 任务执行效率与复杂任务拆解问题** 面对多环节、高复杂度任务，单一 AI 易出现执行卡顿、任务边界模糊、资源分配不合理等问题。本段课程将介绍指令集的设计规范与子代理的划分逻辑，通过指令集实现对 AI 的精准操控与意图传递，通过子代理承接细分任务，形成“主代理统筹 + 子代理执行”的协作模式，大幅提升 AI 处理复杂任务的效率与精准度。	2 小时演示讲解 2 小时动手实验	测评 + 实操作业
模块 6	**MCP 模型上下文协议** 本模块聚焦 AI 与外围研发管理系统的对接核心，通 MCP 构建标准化交互协议，衔接前文指令集与子代理体系，实现 AI 与需求、设计、代码、测试、流水线等研发管理系统的无缝对接，打通 AI 工作流与研发全流程的数据链路。	**解决 AI 与研发管理系统对接适配及数据流转问题** AI 在融入研发全流程时，易面临各类研发管理系统接口不统一、数据格式异构、对接适配繁琐等问题，导致 AI 难以高效获取或推送研发数据。本段课程将解析 MCP 的核心设计逻辑与适配机制，说明其如何通过标准化协议消除不同研发管理系统的对接壁垒，支撑子代理快速调用各系统资源、同步数据信息，让 AI 深度嵌入研发全流程，实现需求拆解、代码生成、测试验证等环节的自动化协同。	2 小时演示讲解 2 小时动手实验	测评 + 实操作业

课程详情

模块	内容	目标	形式	考核形式
模块 7	**CLI （ AI Coding 工具命令行形态）** 本模块聚焦 AI Coding 工具的本地化高效落地，以腾讯 CodeBuddy Code CLI 为核心示例，兼顾 Claude Code 等同类工具，衔接前文 MCP 与研发管理系统对接体系，实现开发者本地环境与 AI 能力、研发流程的无缝融合。	**解决多端 AI 能力调度不一致及本地调用效率问题** 开发者在本地、云端、研发流水线等多端协同编码时，常面临 AI 能力调度标准不一、跨端切换繁琐、能力复用成本高的问题，同时本地编码也存在工具切换打断节奏的痛点。 CLI 作为 AI Coding 工具的命令行形态，核心优势在于提供多端统一调度 AI 能力的方式，本模块将围绕腾讯 CodeBuddy Code CLI 展开，讲解其如何实现本地、云端、流水线等多端环境下 AI 能力的标准化调度，同时覆盖本地代码生成、实时语法校验、一键对接研发流水线等实操内容，兼顾 Claude Code 等同类工具的通用逻辑，帮助开发者在多端场景中高效调用一致的 AI 能力，打通“多端调度 -AI 辅助 - 研发系统同步”的全链路。	2 小时演示讲解 2 小时动手实验	测评 + 实操作业
模块 8	**规约驱动编程 SDD 实战** 本模块聚焦规约驱动编程（ SDD ）的实操落地， GitHub 开源的 Speckit （轻量规约驱动工具包）与 OpenKit （含 OpenSpec 迭代工具，适配全场景研发）为核心示例，衔接前文 AICoding 能力、 MCP 研发系统对接及 CLI 多端调度体系，针对 AI 辅助研发的核心痛点提供解决方案，覆盖 0-1 新项目搭建与 1-N 迭代优化全场景，实现从规约设计、任务拆解到代码生成、验证归档的闭环实战，	**解决 AI 辅助研发中质量不可控、规约与执行脱节及协作低效难题** AI 辅助编码易陷入“氛围编码”（ vibecoding ）的无序状态，核心痛点包括：规范描述模糊导致 AI 生成代码存在隐性漏洞、不符合生产级标准；规约与代码、测试用例异步更新，形成“规约归规约、开发归开发”的落地断层；团队协作中规约理解不一致，版本管理混乱且无明确审计追溯依据；规范不可复用、多 AI 工具适配繁琐，易产生供应商锁定风险。本段课程将结合 Speckit 与 OpenKit 工具特性，讲解 SSD 核心逻辑与实操步骤——依托 Speckit 构建结构化规范、约束 AI 行为，通过 OpenKit 实现规约迭代	2 小时演示讲解 2 小时动手实验	测评 + 实操作业

运营及考核机制（软通内部机制，仅供参考）

此课程由 软通动力金融事业群软件工程事业部（ FINSE ）打造，课程采用 实战演示 + 动手实验的形式

考核标准：考勤积分 *10%+ 测评成绩 *40%+ 作业成绩 *40%+ 学习社区互动 *10%



- 考勤积分：每次考试前，会组织线上打卡签到（前中后），每一次全勤得 10 分；
- 测评成绩：每次训后 3 日内，完成线上测试，每次测评有 3 次机会，满分 100 ， 80 合格；
- 作业成绩：每次训后，需要提交操作作业，专家评估成绩，满分 100 ， 80 分合格；
- 社区互动：参与线上学习内容分享，输出实战案例，均可获得积分， 10 分 / 次；

学习要求：

- 每完成一个模块的学习才能晋级下一个模块；
- 每完成一个模块将收到证书和徽章；
- 每次学员报名，培训组织方需要核对信息，确认无误后进入新的学习群





谢谢