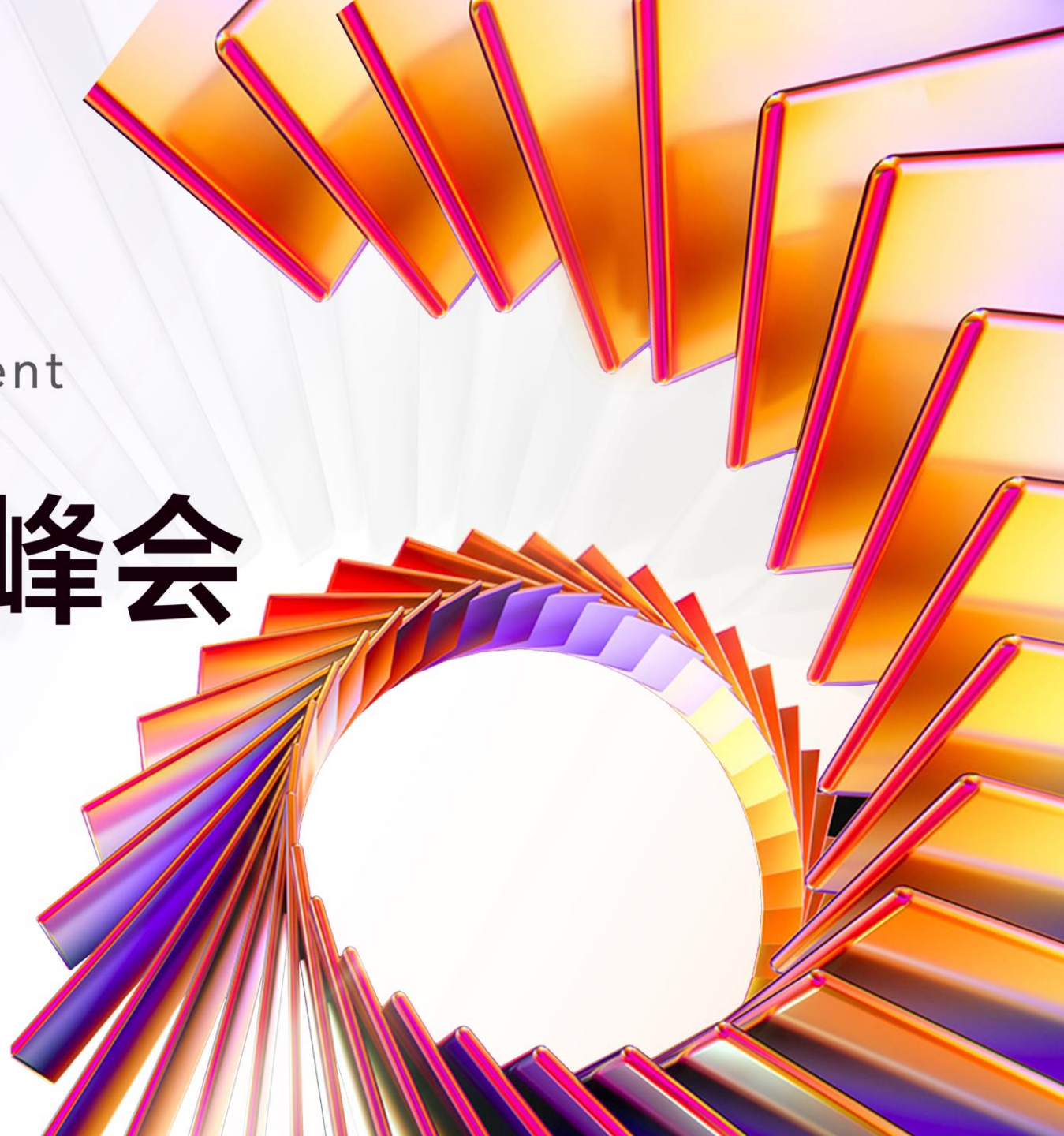
AiDD 7th 2025

第7届 AI+ Development Digital Summit AI+研发数字峰会

拥抱AI 重塑研发

8月8-9日 | 北京站



AiDD 8th 2025

第8届AI+研发数字峰会

拥抱AI 重塑研发 AI+ Development Digital Summit

查看会议详情

下一站预告 11/14-15 | 深圳站 12/19-20 | 上海站

深圳站论坛设置

智能装备与机器人

超越“编程 Copilot”

下一代知识工程

智能网联与汽车智能化

AI 测试工具开发与应用

AI 基础设施和运维

数据智能及其行业应用

可信 AI 安全工程

大模型和 AI 应用评测

多 Agent 协同框架

从智能测试到自主测试

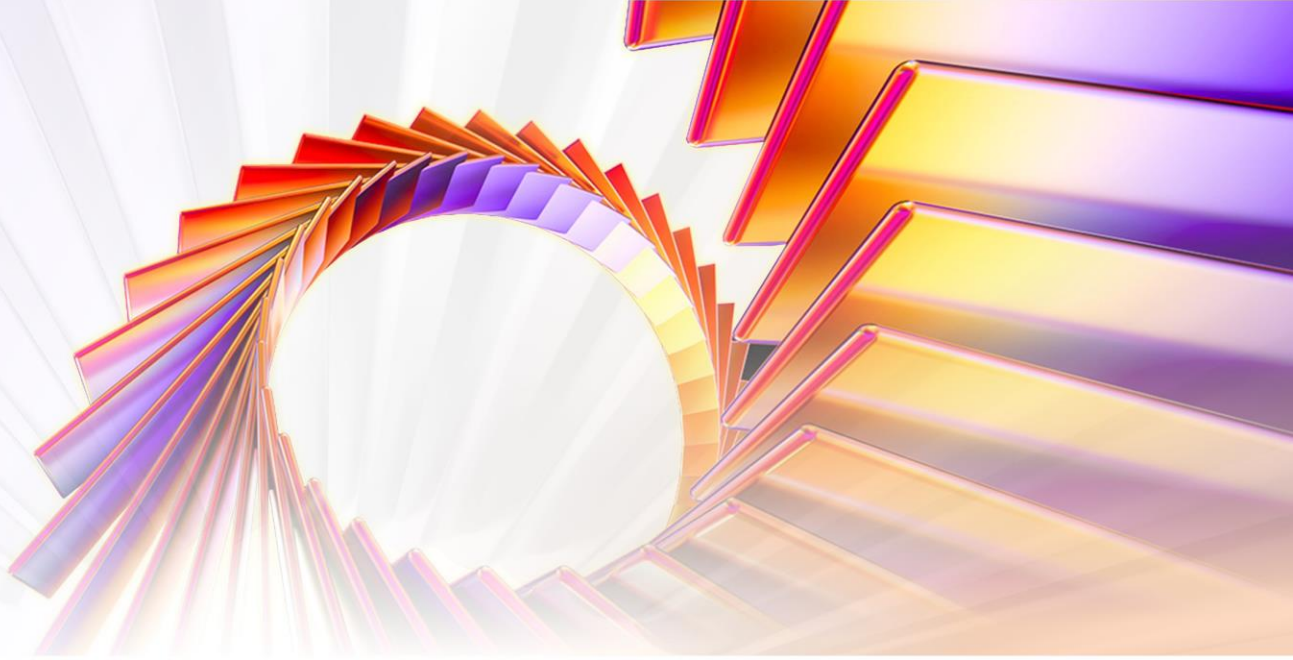
大模型推理优化

多模态 LLM 训练与应用

智能化 DevOps 流水线

上下文工程

AiDD 7th 2025 | 8月8-9日 | 北京站
第7届 AI+ Development Digital Summit
AI+研发数字峰会
拥抱AI 重塑研发



评测驱动开发（EDD）：AI原生飞轮驱动研发效能跃迁

蒋学鑫 | 中兴通讯

蒋学鑫

中兴通讯中心研究院 AI研发提效总体组专家

中兴通讯青年领军人才，软件研发领域资深专家。带领团队研发的操作系统产品曾获第四届中国工业大奖，第21届中国国际软件博览会金奖。曾担任中兴通讯操作系统产品部研发经理、项目经理、副部长等。

目录
CONTENTS

PART 01

AI+时代研发范式的挑战

研发范式与效能

评测的本质

AI原生飞轮

软件工程的演进

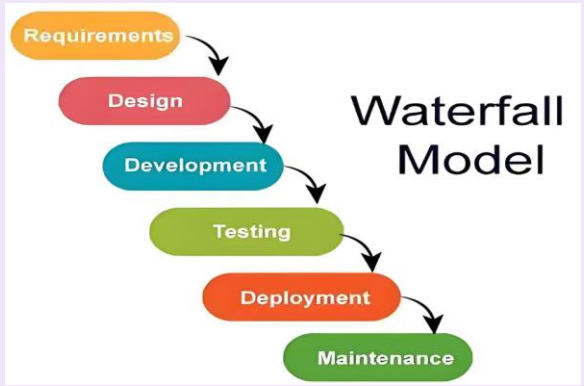
软件工程1.0 | **软件工程2.0** | **软件工程3.0**

1968年
NATO会议
软件工程学科诞生

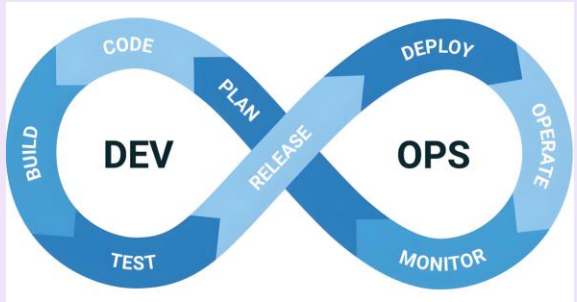
2001年
敏捷宣言发布

2023年
chatGPT/GPT-4发布

- **结构化、规范化、工程化**
- **瀑布模型、V模型**



- **敏捷开发**
- **CI/CD、DevOps**



软件工程3.0宣言 -- 同济大学 朱少民

人机交互智能胜于研发人员个体能力

业务和研发过程数据胜于流程和工具

可生产代码的模型胜于程序代码

提出好的问题胜于解决问题

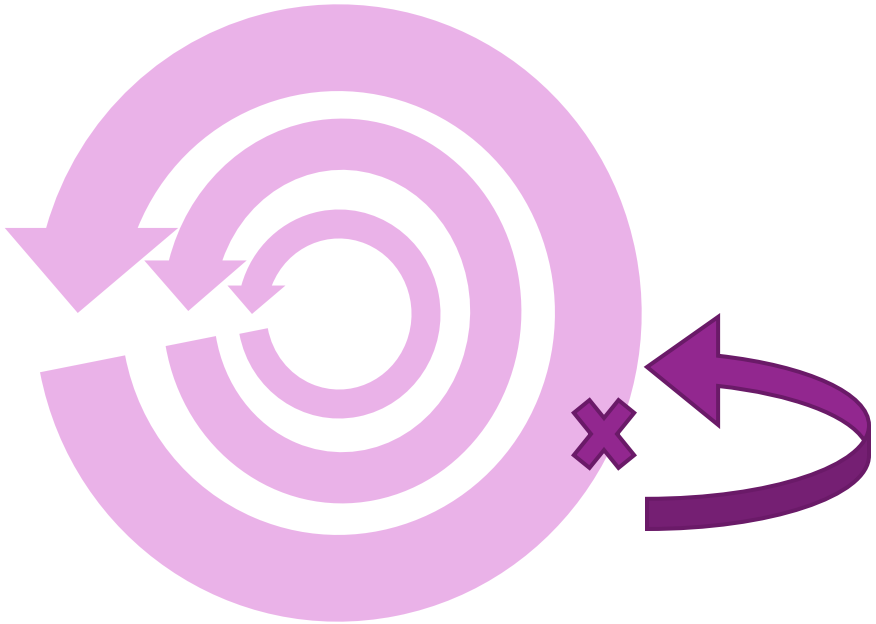
测试模式:	先开发后测试	TDD	?

研发效能的根本瓶颈

- **软件工程1.0（瀑布模型时代）：** 流程刚性导致的变更高成本，瀑布模型的线性开发流程严格限制需求**变更**，且缺陷修复周期长达数周。
- **软件工程2.0（敏捷与DevOps时代）：** 快速迭代很好的解决了1.0时代的需求变更效能瓶颈。但快速迭代发布的质量**风险**，以及质量问题发生后的应对，成为研发效能的隐形瓶颈。
- **软件工程3.0（智能化软件工程时代）：** AI行为不确定性导致产品发布和修复的风险和问题放大。

决定研发效能瓶颈的根本要素是什么？

- ✓ **反馈链路的长短**
- ✓ **反馈链路的畅通性**

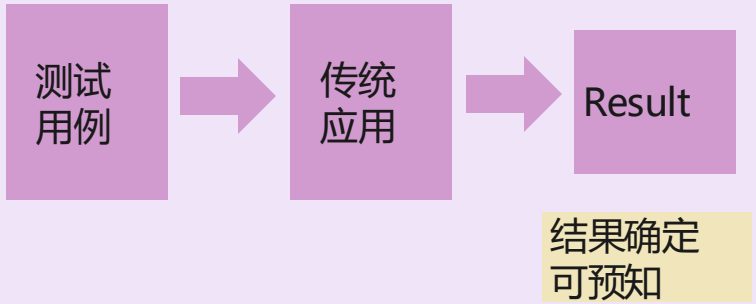


让反馈链路更短，让反馈链路更通畅

测试&评测的本质和目标

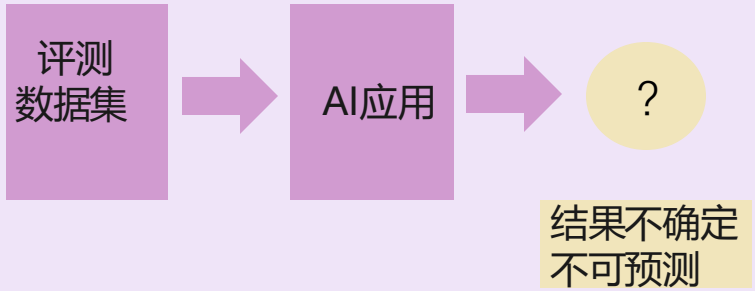
- **测试&评测的本质是反馈，**反馈效果的三要素：准确、全面、高效
- **软件测试的目标是建立对软件可靠性的可控信任，**将软件的不可靠风险控制在可接受范围

传统测试：信任是基于确定性验证。软件行为是确定性的；软件逻辑是显式的



通过“穷尽显式逻辑”实现控制

AI评测：行为是概率性的， 逻辑是隐式的



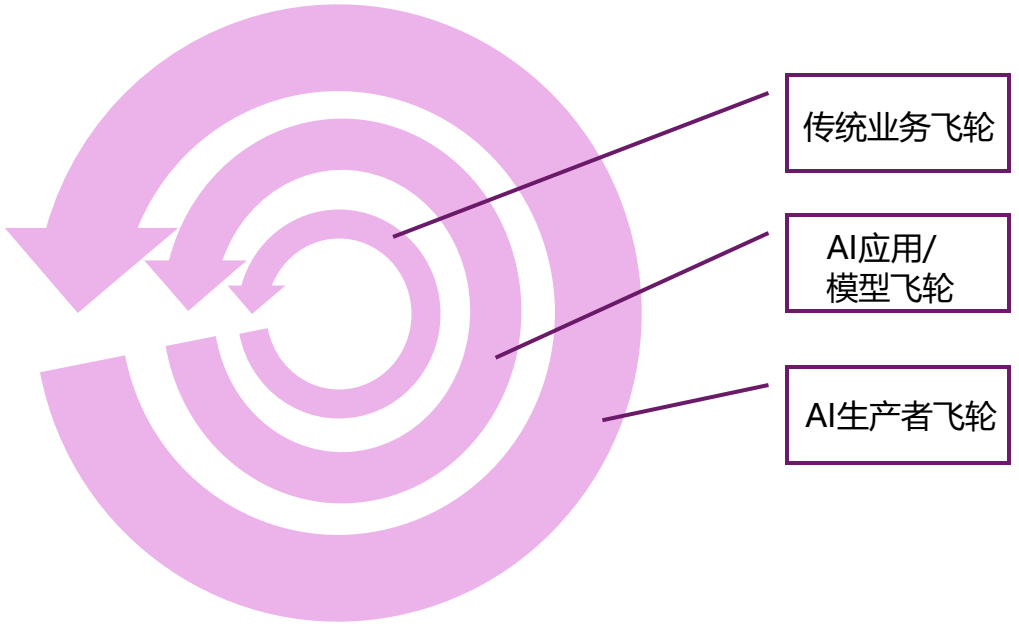
通过“理解隐式概率分布”实现控制

我们要做的就是抓住本质，达成目标

AI原生飞轮

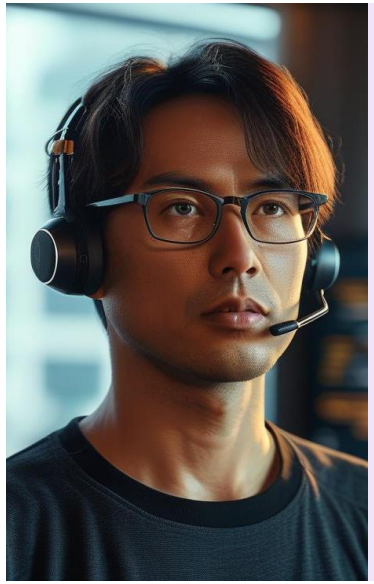
- **AI原生（AI-Native）：**指从系统设计之初即**以人工智能为第一性原理**，重构产品架构、业务流程与组织形态，而非在既有系统上叠加AI功能
- **云原生（Cloud-Native）：**基于云计算特性设计的应用架构方法论，通过容器、微服务等技术实现“为云而生”
- 云原生是**效率革命**，AI原生是**认知革命**

AI原生飞轮（AI-Native Flywheel）是指以人工智能为核心驱动力的自我增强闭环系统，其本质是通过AI生产者、AI应用/模型、传统业务等的数据反馈循环，实现系统的自主进化与价值放大。



通过数据牵引，让不同层面上的飞轮都飞起来

AI软件研发根本性差异--从确定性到概率性



传统软件开发的确定性

- 输入输出关系明确可预测
- 逻辑流程线性可控
- 边界条件清晰定义
- 测试用例覆盖完整

AI智能体的概率性本质

- 输出具有不确定性分布
- 决策过程黑盒化
- 边界模糊且动态变化
- 测试覆盖难以穷尽

方法论变化			
■	可预测性	→	可解释性
■	确定性验证	→	概率性评估
■	静态测试	→	动态评测
■	产物检验	→	能力认证

我们要构建AI时代的软件评测体系

当前评测的痛点和挑战

■ 覆盖不全：评测维度不够完备，难以精准定位系统的短板

- 现有的评测方法通常仅提供单一维度的得分或排名，缺乏对智能体在不同场景下的能力评测
- 评测方式无法为开发者提供具体的优化方向，导致提升智能体能力的过程缺乏针对性

评测集要全

■ 评价差异：端对端能力存在评价差异

- 评测数据孤岛化，不同团队、不同工具的评测结果无法有效整合和对比
- 评测数据和试用主观感受有时出现偏差

评价器要准

■ 反馈缓慢：评测和生产脱节，无法快速应用评测结果

- 缺乏持续改进的闭环机制，导致AI产品能力提升缓慢，用户体验改善不明显
- 缺乏自动化和高效的评测工具使得新版本的效果验证耗时较长， 影响了智能体在生产环境中的快速部署和应用，延缓了产品迭代的速度

评测反馈要快

需要解决当前评测面临的完备性、客观性、时效性三大挑战，让反馈更加全面、准确、高效

PART 02

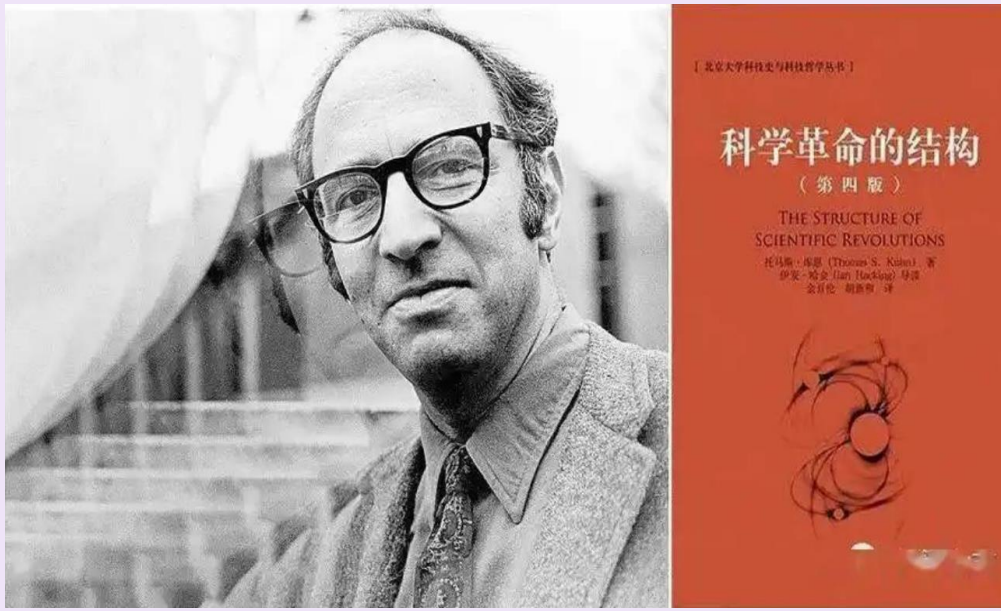
基于EDD的整体解决方案

EDD的概念

- **EDD（Eval-Driven Development）是一种以持续评测为核心驱动力，通过系统性验证和优化AI智能体的内在能力与行为逻辑，确保其输出物满足质量要求的方法论**
- **EDD本质是将评测机制深度嵌入全流程，形成“评测-反馈-优化”闭环，是AI时代软件评测的“新范式”**

- **“范式”核心定义是“共享认知框架”**
  - 由科学哲学家托马斯·库恩提出，包含三个层级： 本体论、认识论和方法论，这三个层级构成了一个“认知坐标系”
  - 本体论决定了“研究本质是什么”，认识论决定了“研究标准是什么”，方法论确定了“实现路径是什么”

- **“新范式”的本质是“认知框架的重构”**
  - 传统测试范式的核心是“用人类可理解的逻辑驯服软件的复杂性”，而大模型的出现让“复杂性”突破了人类的认知极限
  - 新范式的意义不在于“找到更好的测试工具”，而在于承认“人类无法完全理解软件的工作原理”，并建立一套基于概率、涌现性和动态适应的新认知框架



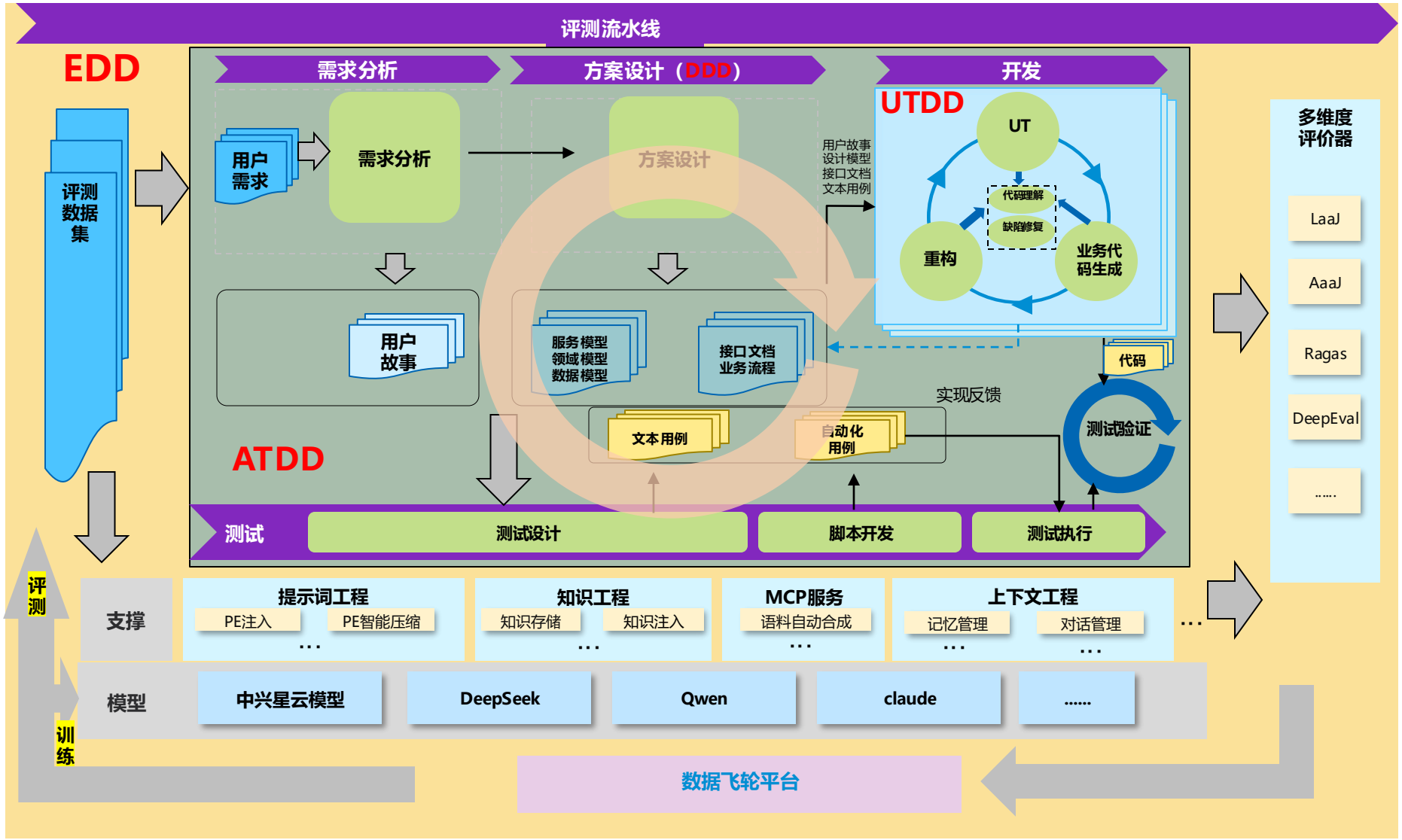
EDD是AI时代软件评测的“新范式”

EDD适用的对象

AI产物	**AI产物有三种模式** • **Embedding嵌入模式：**大模型被集成到现有的应用程序或服务之中，为用户提供智能和个性化的体验；不直接暴露模型能力给最终用户，而是将其作为一种增强的功能或服务的一部分。 • **Copilot助手模式：**通过AI技术赋能的智能助手，协助人类完成各种任务；在特定领域提供帮助，通过与人类交互来提高效率和创造力。 • **Agent代理模式：**智能实体，能够感知环境、进行决策和执行动作；具有自主性，能根据给定目标或任务独立进行规划、执行和反思；在复杂性和自主性方面更为先进。
AI生产者	**软件工程3.0模式：**AI技术广泛应用于软件开发的各个环节，改变了软件开发的范式和流程，成为独立的或与人类协同的软件生产者。以Copilot助手和Agent代理模式为主，交付物为软件工程过程产物。

AI产物也可能是AI生产者， AI可以制造AI

EDD与传统xDD：从验证产物到优化生产者



- **DDD（领域驱动设计，Domain-Driven Design）：** 解决业务复杂性，通过领域建模构建可持续演进的架构
- **UTDD（单元测试驱动开发，Unit Test Driven Development）：** 从代码层驱动开发，确保微观质量
- **ATDD（验收测试驱动开发，Acceptance Test Driven Development）：** 从业务层驱动开发，确保宏观价值

EDD：从验证产物本身到优化生产者（模型、智能体、支撑工具等）

EDD与TDD的异同

- **TDD是工业时代软件的“精准校准逻辑”，**通过精准测试用例确保每个部件的行为符合设计
- **EDD则是大模型时代的“系统评估逻辑”，**通过持续测评把握其能力边界和风险，再反向驱动开发

两者的核心逻辑一致

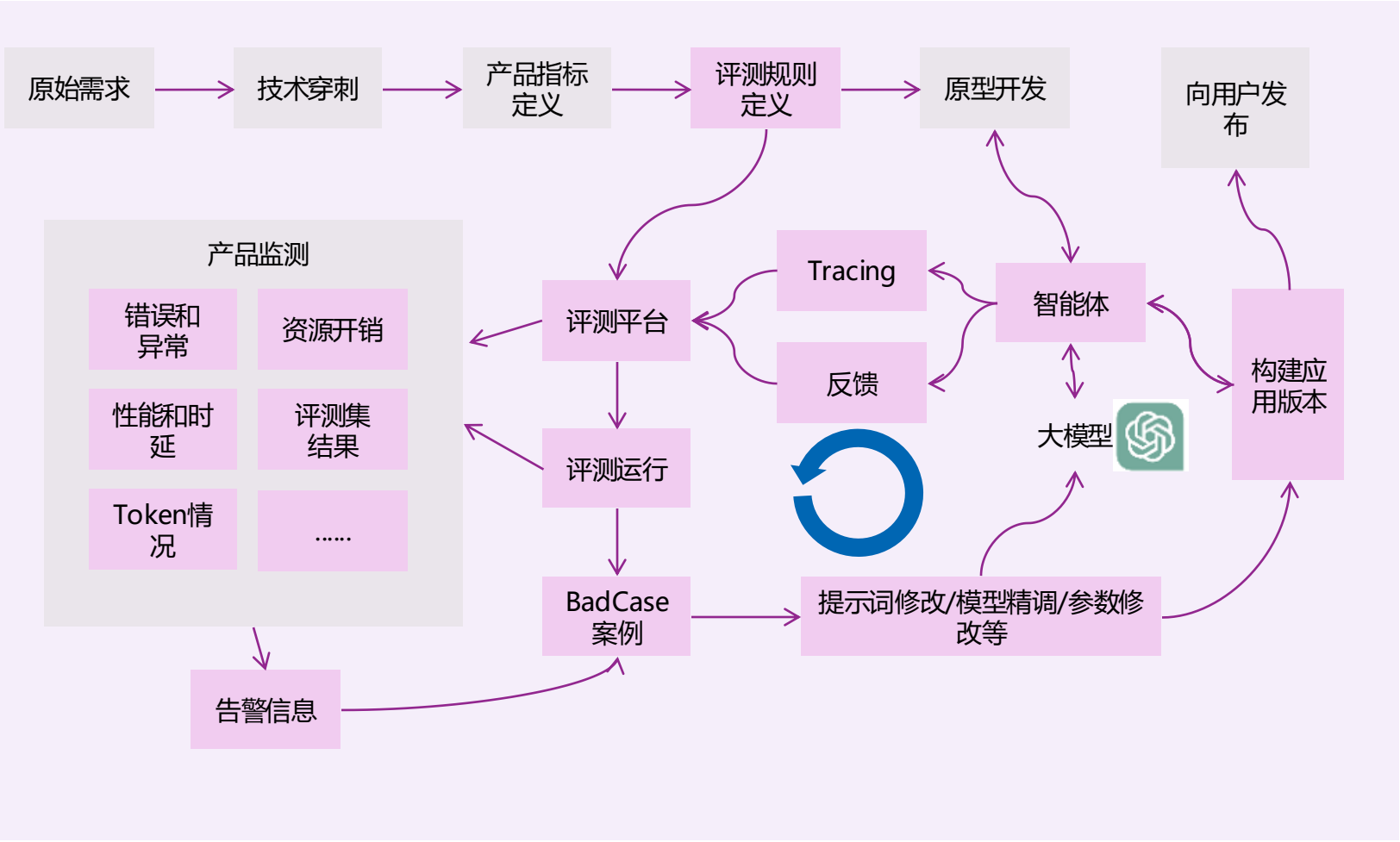
- 遵循“先定义验证标准，再进行开发实现”的闭环，将“质量验证”前置到开发流程中，而非事后补救。
- 本质是对“开发-验证”关系的重构——不再是“先做出来再检查”，而是“明确合格标准后再动手”，通过迭代验证倒逼开发质量。

两者有三点不同

- **对象不同：**TDD的测试对象是结构化、确定性的软件模块，比如函数、接口、业务逻辑。这些模块的行为是“可拆解、可预期”的，测试用例可以精准覆盖所有分支，理论上覆盖率可以做到100%。EDD的测评对象是大模型、智能体等涌现性系统。其核心是不可穷尽的涌现能力。
- **标准不同：**TDD通过的需要把一条条静态的用例pass，是一个明确的阈值，验证的是代码功能正确性。 EDD验证的是 行为分布可靠性，怎么算通过，是一个多维的复杂的标准。
- **方法不同：**TDD的测试方法是用例覆盖，通过设计足够多的测试用例如单元测试、集成测试、验收测试等，尽可能覆盖代码的所有分支和场景，追求“测试用例通过即合格”。这种方法的前提是“场景可穷尽”。EDD的测评方法是多维动态评估，由于大模型的场景不可穷尽，且行为具有概率性，无法通过“覆盖率”验证。

从TDD到EDD，是验证范式的“扩容”和“升维”

基于EDD的AI产品研发流程



- 从用户需求进行技术穿刺、产品指标定义是传统流程
- **评测规则定义**是整个流程中关键一环，需要针对智能体系统拓扑结构中的任何节点都应该准备好评估数据集：输入→预期输出
- 在原型开发中，需要**提前考虑埋点**，使得整个系统具备更细粒度的可观测性
- **可观测性平台**能帮助实现高效搜索与可视化，同时支持快速版本迭代并添加自动化评估能力。
- 聚焦未通过的评测集，对相应系统给出**反馈和修正举措**
- 构建版本对用户发布的同时，**EDD循环持续进行**，不断完善产品
- 在整个过程中，内部细化信息通过监测**向用户告警**

EDD贯通AI产品全生命周期流程

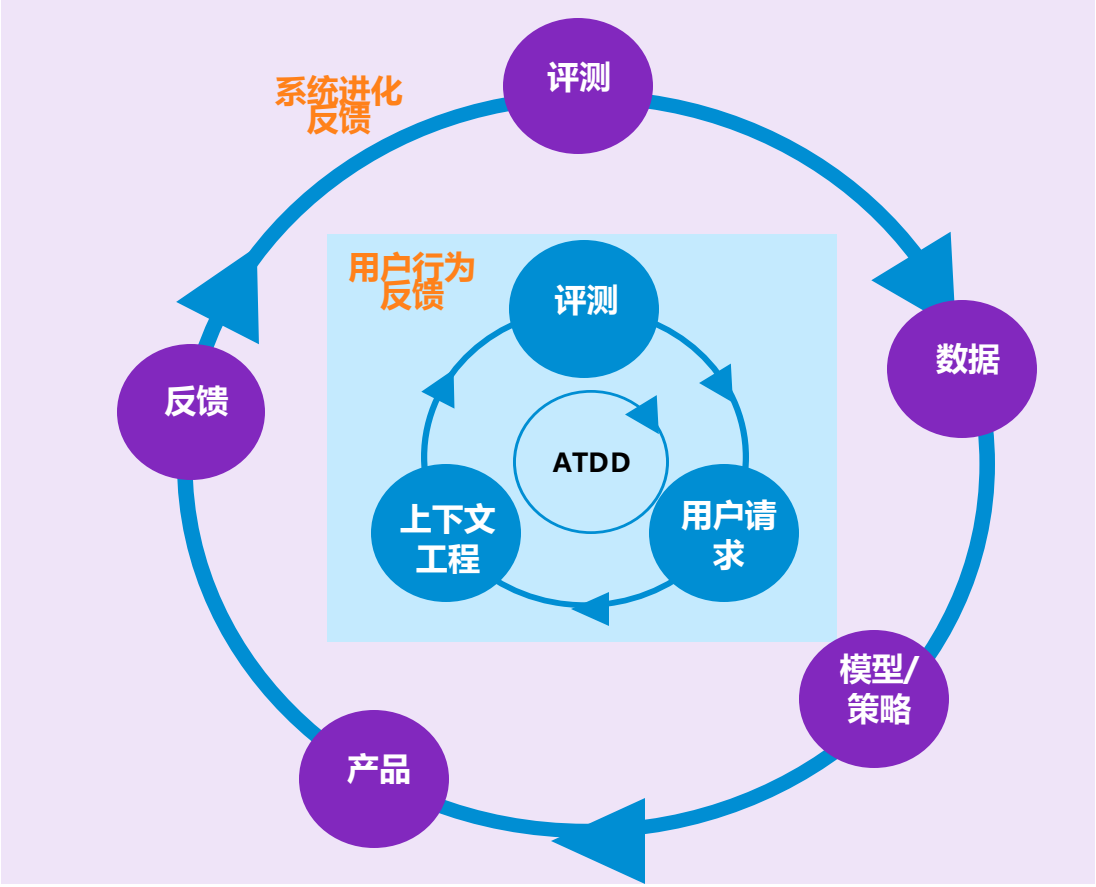
不同模式评测关注点

AI模式	特点	评测关注点	评测维度
Embedding嵌入模式	对系统来说，相当于嵌入了一个高度不可信任的子模块	输入输出不确定性对上下游模块的影响；（系统设计时，借鉴工程控制论思想，将AI不可靠因素控制在一个确定范围）	ATDD验收标准 错误处理
Copilot助手模式	✓ 自主性弱，依赖人工输入和指导 ✓ 只在特定领域提供辅助 ✓ 基于已有知识和模式提供建议	✓ 对人工指令的理解和响应能力 ✓ 在特定领域任务中的辅助效果 ✓ 在特定领域中的深度和精度	成功率、稳定性、性能评测
Agent代理模式	✓ 自主性强，能主动感知环境，制定决策并执行任务 ✓ 可将复杂任务分解成多个子任务并规划执行 ✓ 从经验中学习和自我优化，适用于多种复杂场景和领域	✓ 无人类干预下自主完成任务的成功率、效率及质量； ✓ 复杂任务的分解和执行能力； ✓ 不同领域和场景下的通用性和适应性；	以准确率、采纳率、稳定性和性能评测为主

根据AI不同应用模式有针对性的进行评测关注

评测反馈闭环

通过数据驱动的评测闭环系统，将应用验收、用户反馈、模型优化等环节有机整合，形成多维度自我强化的增长循环，实现系统的自主进化与价值放大



多维反馈环：通过构建**具有学习特征的记忆能力**等实现**多维反馈，在不同层次上**都能够**自主进化**，提升适应性

- **用户行为反馈：**提升用户的指令驾驭能力，基于程序/语义/情景记忆通过上下文工程分析并评估用户用法，增强体验

- **系统进化反馈：**通过评测结果，反向作用智能体和大模型的进化
  - 智能体自进化反馈：增强同类任务规划与生成能力，从用户历史轨迹中提炼最佳实践动态优化提示词
  - 大模型自优化反馈：提升大模型的短板能力，通过数据飞轮从历史轨迹中合成正反向语料

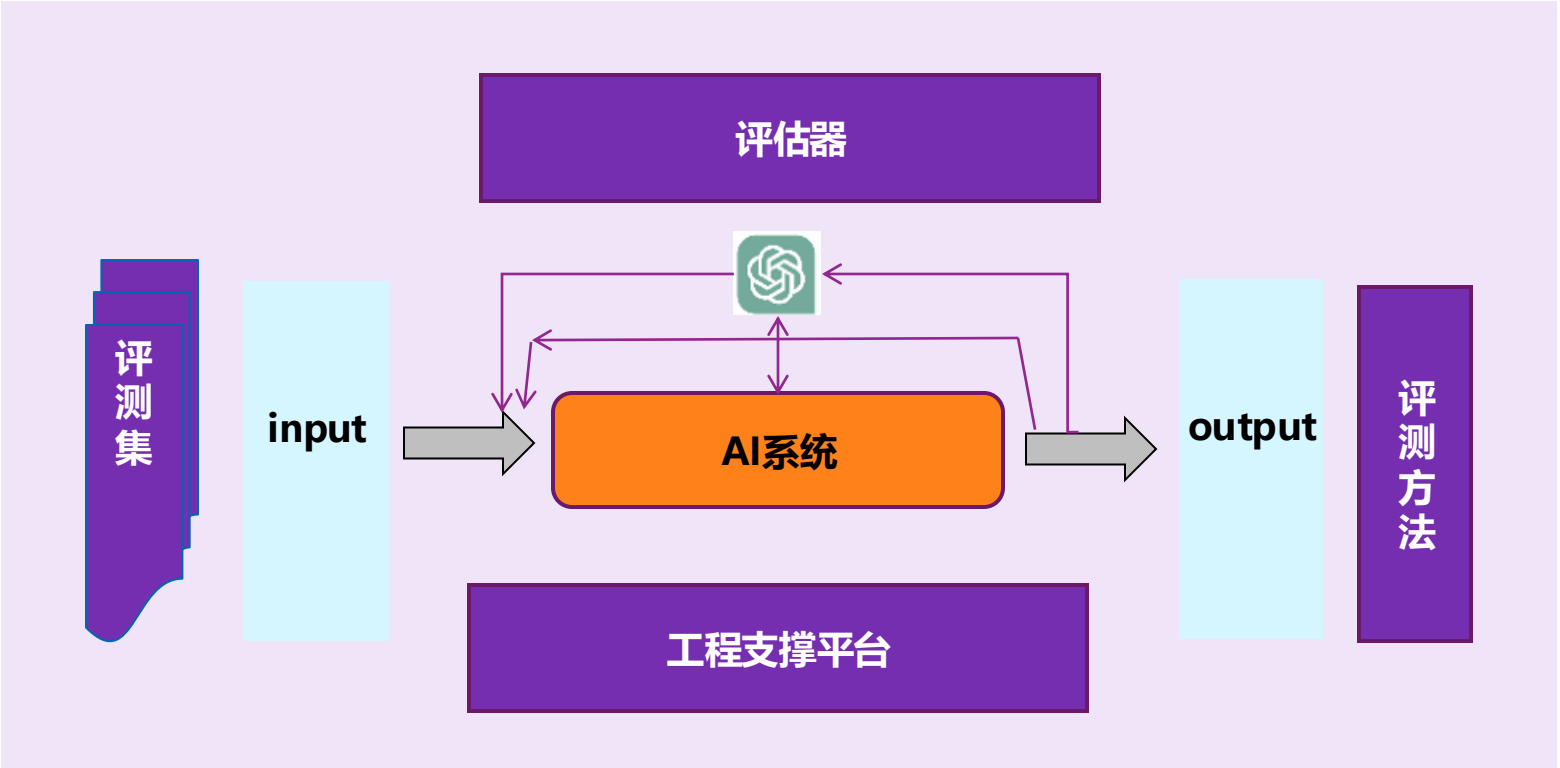
多层次的评测反馈闭环，形成AI原生飞轮

PART 03

企业EDD的具体落地实践

企业EDD实践主要举措

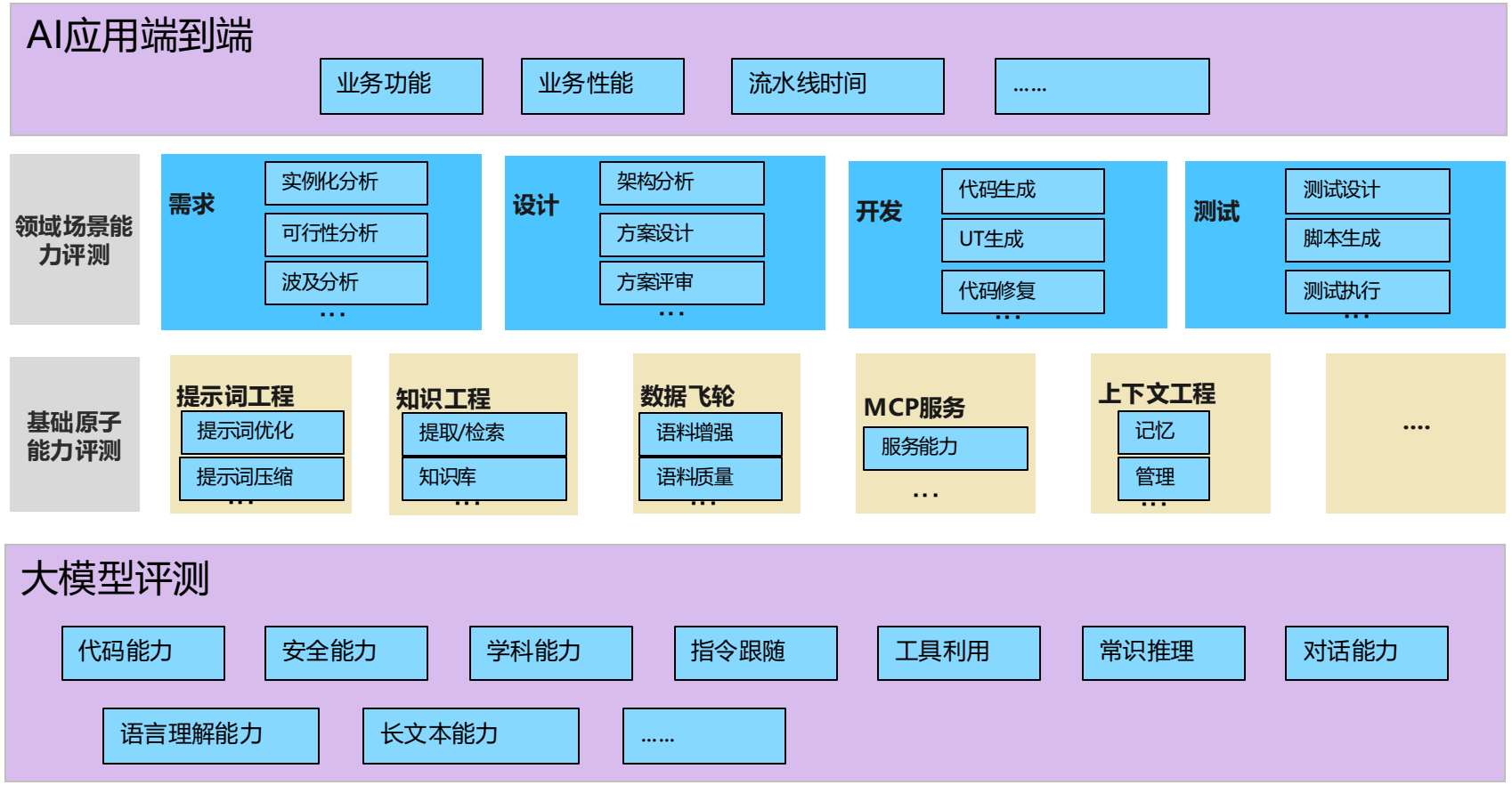
EDD实践主要从评测集、评估器、评价方法、工程支撑平台四个方面展开



- **评测集：**基于历史采样、用户日志、合成数据等方式产生。合成数据可以采用人工、工具、LLM等方式生成
- **评估器：**包括人工评估、LLM-as-a-Judge、Agent-as-a-Judge、启发式等方式判决
- **评测方法：**包括线上评测和线下评测两种场景
- **工程支撑：**包括流水线和数据分析、告警平台等

构建多层次的评测集全景

评测不仅是针对大模型，而是覆盖大模型、基础原子能力、领域场景能力和业务端到端能力的多层次多维度的评测



静态评测集大致策略：

- **应用端到端**以自研评测集为主，开源评测集为辅，更能反映场景真实情况
- **领域场景能力**以自研测评集建设为主，可更好的反映出业务场景的生成效果情况
- **基础原子能力**以开源测评集为主自研评测集为辅，用统一的尺子去衡量基础原子能力工具在业界的水平
- **大模型自身能力**维度测评用业界成熟测评体系覆盖，以开源评测集为主自研评测集做辅助

评测集核心要素

覆盖全面

- 全场景分析，立体标签体系，贴近用户真实输入输出
- 通用评测和场景评测集相结合
- 注意隔离、更新和噪声消除

多路来源

- 人工合成数据：按照既定格式要求手工收集的数据
- LLM生成数据：使用LLM生成特定数据
- 开发工具生成：使用工具/脚本生成，或在业务生产过程中自动生成
- 用户使用日志：使用生产环境真实数据进行回测
- 开源benchmark：使用业界通用开源评测集评测

版本管控

- 评测集作为重要生产资料，以版本形式管控
- 按策略更新评测集，并记录变更日志，确保评测集的时效性
- 收集实际使用中的反馈，不断调整和优化评测集

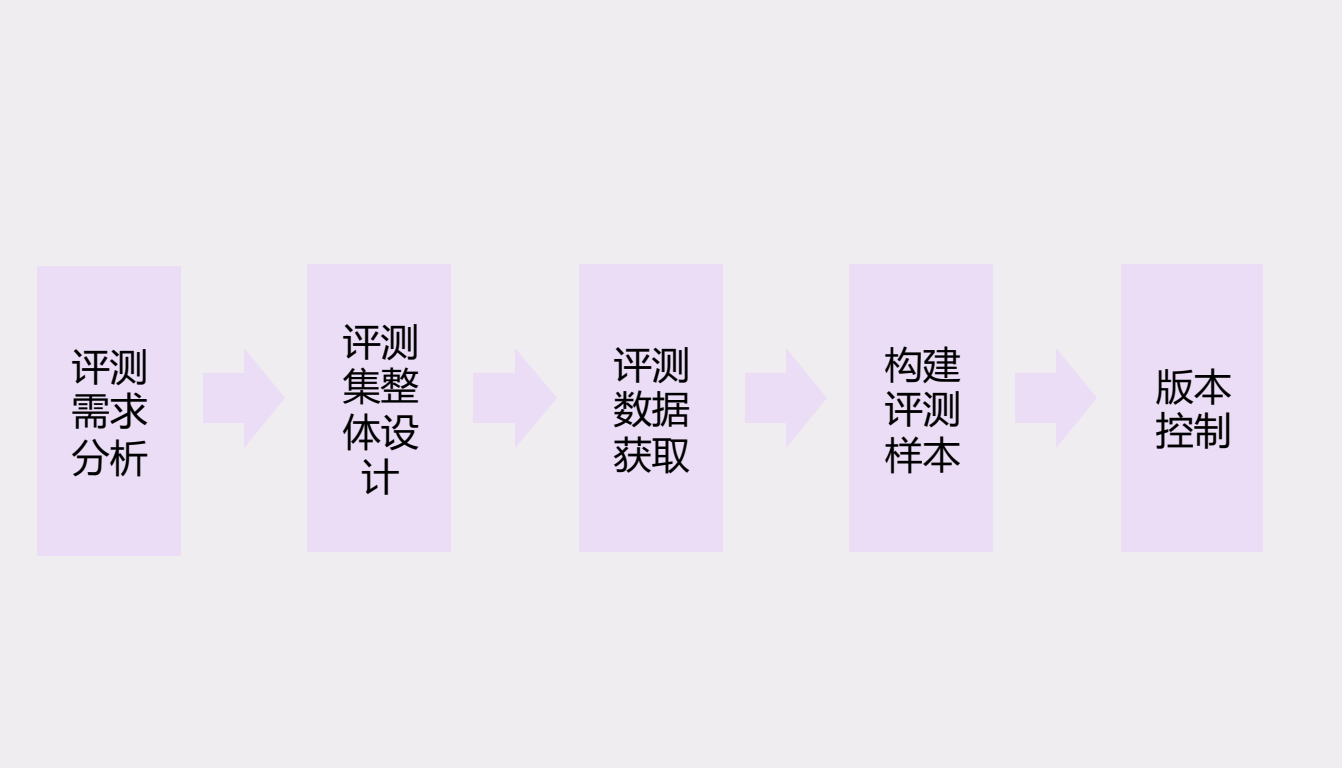
静态benchmark和动态用户场景评测集的部署，在不同阶段有不同策略：

- **开发阶段：**这个阶段关注基础能力锚定， 以benchmark为主，通过标准化测试快速定位模型缺陷，确保基础能力达标 。
- **小规模部署阶段（灰度、FOA等）：**关注场景化能力映射，以动态数据权重 ，这个阶段，应用范围较小，噪声相对较小，可以用真实数据充分验证场景能力。
- **规模阶段：** 需要混合权重分配，并根据反馈动态调整比例。注意语料的处理，去除噪声，得到真实有效的数据。

评测集建设的关键是数据能够代表实际应用中将要面对的真实场景分布

评测集构建方法

评测集全生命周期管理



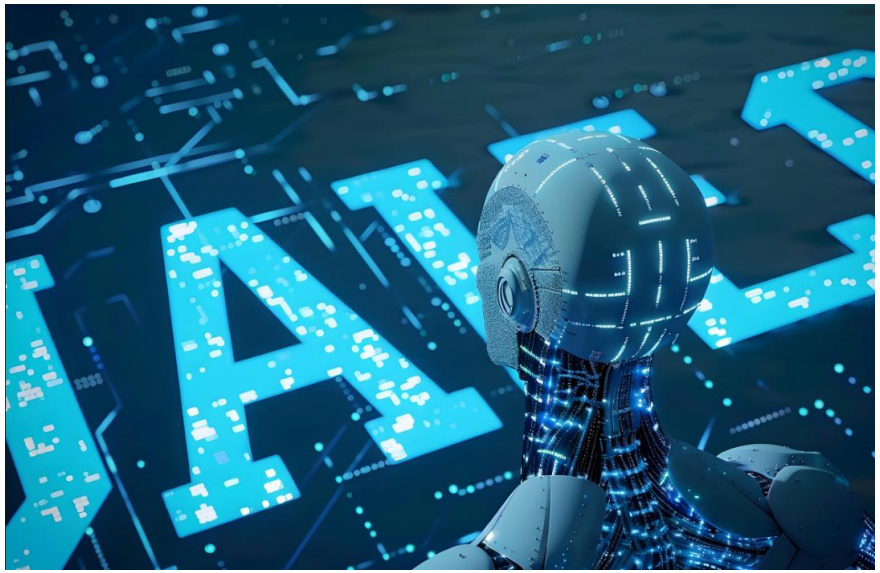
示例：精细化场景的评测集覆盖示例

领域	场景	说明
需求智能	需求准入	需求提交需要按照标准格式模板提交，这里需求准入是指对需求提效的规范性等做
	实例化分析	目的主要进行用户需求的价值分析，使用实例化的方法进行用户的价值分析，也可
	可行性分析	目的是评估用户需求在产品内部是否可以实现，通常会先做协议分析、标准分析、
	波及分析	进行市场需求的波及分析，用于分析当前市场需求波及到的特性、用例或系统组件
	工作量评估	目的是为了进行项目需求分发的决策，需要对新增的需求进行工作量投入的评估。
	需求分析评审	检查需求文档的完整性、一致性和可行性，确保高质量的需求输出。
设计智能	架构分析	对拟设计的产品架构进行架构对标和架构决策，形成关键架构需求以指导后续架构
	方案设计	在系统架构设计的基础上，针对单条或多条功能特性进行规划和定义，以指导特性
	方案评审	智能化评审设计方案，提供改进建议和风险评估。
	架构重构	自动识别架构中的问题并提供重构建议。
	架构治理	持续监控和优化架构，确保系统的健壮性和可扩展性。
	领域建模设计	自动化领域建模，生成符合业务需求的模型。
	4A架构设计	采用4A（业务架构、应用架构、技术架构、数据架构）的全面的架构设计方法
开发智能	代码生成	基于需求和设计，自动生成代码。
	代码补全	智能代码补全，提高开发效率。
	代码注释/代码解释	自动生成代码注释和解释，提升代码可读性。
	代码优化	自动识别和优化代码中的性能问题。
	代码纠错	自动检测和修复代码中的错误。
	代码评审	智能化代码评审，提供改进建议。
	UT生成	自动生成单元测试用例，确保代码质量。
	代码检查	静态和动态代码检查，发现潜在问题。
	代码翻译	将代码翻译成不同的编程语言。
	FT生成	自动生成功能测试用例，确保系统功能完整。
	代码调试	智能化代码调试，快速定位和修复问题。
测试智能	测试策划	自动生成测试计划/测试策略，确保测试覆盖全面。
	测试分析和设计	基于需求文档进行测试分析，生成测试用例。
	测试脚本开发	自动生成自动化测试脚本，提高测试效率。
	测试用例评审	对测试用例/自动脚本进行智能评审
	测试执行结果分析	智能分析测试执行结果，提供改进建议。
	测试用例调度	智能调度测试用例，优化测试资源。
	测试故障定位	自动化故障定位，快速识别和解决测试问题。
	测试报告生成	自动生成测试报告，提供详细的测试结果和分析。

评估器--裁判模型&裁判智能体

为什么有了LLM-as-a-Judge，还需要Agent-as-a-Judge?

- ✓ 给裁判更多道具： 记忆、工具、专家协同
- ✓ 应对更加复杂的场景



如何让LaaJ&AaaJ更准确?

- ✓ **粒度更细**：一次性只从一个最小的维度进行评判
- ✓ **专用模型**：针对细分场景，训练和微调更专业的裁判模型
- ✓ **交叉验证**：多模型交叉验证、动态评分校准，设定偏差阈值超过时人工介入

评测平台

建设一个好的评测平台，需要具备下面的几大功能

- **评测集管理：**评测任务的执行数据，可管理不同场景、不同版本的评测数据
- **评测流水线管理：**评测任务的执行载体，登记注册评测流水线
- **评测指标&算子管理：**评测数据的分析基准，定义并管理各类评测指标和指标的计算方法
- **评测任务管理：**评测执行的最小单元，支持单任务评测、场景评测、定时评测等多种模式
- **评测报告管理：**评测数据的呈现方式，梳理评测过程数据，汇总评测指标并提炼评测报告
- **统一度量看板：**多维度的度量看板，统计全面的使用情况和各管理功能的数据分布情况

欢迎使用 iEval-AI 评测平台

一站式 AI 评测与管理解决方案

EDD带来研发效能提升

EDD助力零号员工端对端研发提效

- 小反馈环：通过AI自主模式的用户需求分析、设计、开发、测试，交付效率大幅提升
- 中反馈环：通过数据飞轮动态反馈和积累语料，使得模型迭代发布周期大幅缩短
- 大反馈环：通过全场景动态语料的收集、清理和反馈，精准识别智能体生产者能力短板，快速提升智能体能力

PART 04

总结与展望

总结

评测的本质是**反馈**，反馈效果的三要素：**准确、全面、高效**

提升研发效能的关键是**让反馈链路更短，更通畅**

评测数据的关键是能代表在实际应用中要面对的**真实场景分布**

AI原生飞轮通过AI生产者、AI应用/模型、业务程序等的数据反馈循环，实现AI系统的**自主进化**与价值放大

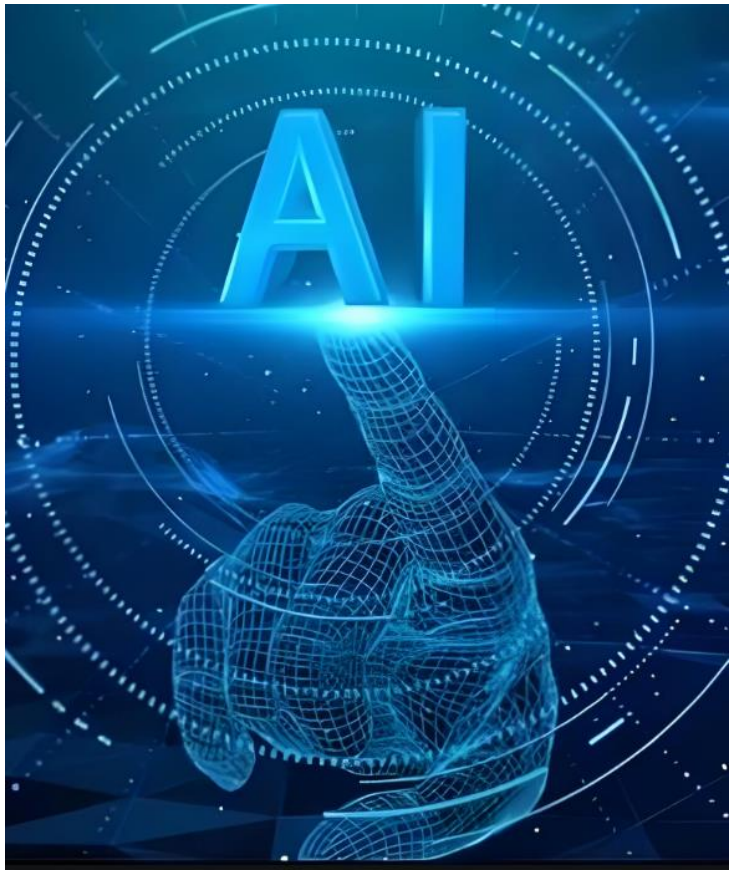
评测通过的标准要**警惕为追求高评估通过率而优化**。如果系统能 100%通过评估，很可能说明设置的挑战性不足

展望

软件工程3.0宣言将快速的深入落地，并持续演进：

- ✓ *人机交互智能胜于研发人员个体能力*
- ✓ *业务和研发过程数据胜于流程和工具*
- ✓ *可生产代码的模型胜于程序代码*
- ✓ *提出好的问题胜于解决问题*

从人机交互到AI主导；数据持续愈加重要；从生成代码的模型到全流程贯通的智能体；定义问题是未来程序员关键能力。



对于程序员来说，需要完成两个关键转变：

- 转变一：从解决问题到定义问题的转变。定义问题就要结合场景深入分析，解决问题交给EDD反馈环。
- 转变二：从训练思维到产品思维的转变。训练思维是以打榜提分为主要目标，产品思维则更多要专注用户体验，AI原生飞轮以产品化为根本目的。

这两个转变，是EDD的重点建设方向。即把评测融入到AI应用/产品的研发过程中，和AI原生紧密协同，才能最大发挥评测的重要作用。

Q&A

AiDD 8th 2025

第8届AI+研发数字峰会

拥抱AI 重塑研发 AI+ Development Digital Summit

查看会议详情

下一站预告 11/14-15 | 深圳站 12/19-20 | 上海站

深圳站论坛设置

智能装备与机器人

超越“编程 Copilot”

下一代知识工程

智能网联与汽车智能化

AI 测试工具开发与应用

AI 基础设施和运维

数据智能及其行业应用

可信 AI 安全工程

大模型和 AI 应用评测

多 Agent 协同框架

从智能测试到自主测试

大模型推理优化

多模态 LLM 训练与应用

智能化 DevOps 流水线

上下文工程

AiDD

「深行·浅智」

Walk Deep, Think Light.

2025.11.16

AiDD首届麦理浩径徒步

KEYLINK ing

科技生态圈峰会 + 深度研习

——1000 + 技术团队的共同选择

AiDD峰会详情

上海站

AI+研发数字峰会

AiDD 峰会

时间：

2025.05.23-24

北京站

AI+研发数字峰会

AiDD 峰会

时间：

2025.08.08-09

深圳站

AI+研发数字峰会

AiDD 峰会

时间：

2025.11.14-15

香港站

「深行·浅智」

AiDD 首届麦理浩径徒步

时间：

2025.11.16

上海站

AI+产品创新峰会

AiDD × 联合峰会

时间：

2025.12.19-20

第7届AI+研发数字峰会
AI+ Development Digital Summit

感谢聆听！

扫码领取会议PPT资料