

THEORETICAL REFERENCE FRAMEWORK

Course as Code

课程即代码

从资源共享走向生态协同的工程化范式

定义 (Definition)

将课程重新理解为一个可版本化、可协作、可治理的知识系统。课程不再是静态的教学内容集合，而是通向开放知识系统的工程化切入口。

TRADITIONAL

内容托管

Content Hosting



COURSE AS CODE

系统工程

System Engineering

核心设计原则 (Design Principles)



单一信任源 (SSOT)

课程必须存在唯一权威源仓库。发布到LMS的内容只是源仓库在特定时间点的确定性投影。



关注点分离

语义核心（教学意图）与表现形式（排版样式）解耦，实现课程的长期可移植性。



课程即有向无环图 (DAG)

显式表达知识对象间的依赖关系，而非隐性的章节顺序，支持依赖分析与重组。



协作即同行评审

课程协作被制度化 Pull Request 与 Review 过程，将分布式判断转化为系统共识。

抽象参考架构 (Abstract Architecture)



代码化表达谱系 (X as Code Spectrum)

Docs & Knowledge as Code

不仅是文本，而是结构化的知识对象。

```
KnowledgeUnit:
  id: KU-Entropy-01
  type: Concept
  prerequisites:
    - KU-Probability-Intro
  content: Reference → ./entropy.md
```

Assignment & Workflow as Code

将“目标是什么”和“凭什么算完成”显式化。

目标 交付物 评价标准

Governance as Code

治理不是“谁说了算”，而是“在什么条件下可以做什么”。制度成为系统的一部分。

社会化协作模型 (Contributor Model)



维护者 (Maintainer)

负责课程结构稳定、合并变更与发布版本。决策权的最终承担者。



贡献者 (Contributor)

通过 Fork 和 PR 提交改进建议。多样化视角的引入者。



使用者 (Consumer)

课程的直接用户，潜在的未来贡献者。反馈循环的起点。

Key Takeaway

课程即代码并非终点，而是通向开放知识生态系统的中介范式。它为构建可演进、可继承、可治理的知识体系奠定了工程化基础。