

过程模型名称	指导思想	特点	应用场景	重量级	难度	优点	缺点
瀑布模型	瀑布模型是最早的软件开发模型之一，采用线性顺序开发，即各阶段的输出是下一阶段的输入。项目的每个阶段严格按顺序执行，不能回溯。	阶段性流程（需求、设计、实现、测试、维护）。 强调文档化。 每个阶段必须完成，才可进入下一阶段。	适用于需求明确、变动较少的项目，比如政府项目、传统软件开发。	重量级。	难度较低，容易理解和实施。	简单易懂。 流程规范，容易管理。 清晰的阶段划分，方便跟踪进度。	不适合需求频繁变更的项目。 缺乏灵活性，无法适应动态环境。 回溯修复成本高。
原型模型	通过创建软件的早期原型，向用户演示并收集反馈，从而逐步完善需求，直到开发出最终系统。	快速生成原型。 基于用户反馈不断迭代。 有助于解决需求不明确的问题。	适用于需求不明确，用户期望快速验证想法的项目，例如用户交互界面设计、创新型产品开发。	中等重量。	难度中等，需频繁与用户互动和调整需求。	能够快速响应需求变化。 增强用户参与感，减少需求误解。 降低开发风险。	可能导致需求膨胀。 原型过多迭代导致时间和资源浪费。
螺旋模型	结合瀑布模型的系统性和原型模型的迭代性，螺旋模型采用多次迭代，逐步细化系统。每次迭代都经历规划、风险评估、开发、用户反馈等阶段。	以风险为驱动，强调风险管理。 包含多个迭代周期。 每个迭代中都会进行风险分析、评估和解决。	适用于复杂、风险高的项目，如国防项目、大型企业系统。	重量级。	难度较高，尤其是风险评估和管理部分。	能够有效处理高风险项目。 提供持续的反馈和改进机会。 更灵活，适应性强。	成本较高，过程复杂。 需要专业的风险管理经验。

Scrum 模型	Scrum 是一种敏捷开发框架，强调自组织团队和迭代开发。工作通过称为“冲刺”的短周期完成，团队通过每日站会沟通进展，持续交付可用的软件。	迭代开发，每个冲刺通常为 2-4 周。 强调团队协作和自我管理。 持续交付，频繁发布。	适用于快速变化和 需求不稳定的项目，尤其是互联网、移动应用开发等领域。	轻量级。	难度中等，强调团队协作和敏捷性。	高效应对需求变化。 透明化管理，鼓励团队自我组织。 快速交付可用产品，获得反馈。	对团队的自律性和沟通能力要求高。 可能导致过度依赖短期目标，忽略长远规划。
极限编程 XP	极限编程是一种敏捷开发方法，强调代码质量、持续集成、反馈、简单设计和频繁发布。其核心理念是以快速迭代、小步前进为原则，在不断反馈中调整开发。	强调代码质量（如结对编程、单元测试、重构）。 持续集成和发布。 简单设计和快速反馈。	适合高变化、高复杂度的项目，尤其是小型团队和互联网软件开发。	轻量级。	难度较高，开发人员需要有较强的技能和自律性。	提高代码质量，减少后期维护成本。 强调用户反馈，快速响应变化。 简单设计，避免过度设计。	对开发团队要求高，尤其是代码质量和团队协作。 不适合大型团队或高风险项目。