# CMMI 成熟度模型评估与软件过程改进计划

## 一、CMMI 层次成熟度模型简述

CMMI（能力成熟度模型集成）通过五个成熟度层次描述组织过程能力的演进，从无序到优化的发展路径如下：

### （一）初始级（Level 1）

* **特征**：过程无序，依赖个人经验，项目成功取决于核心成员能力。
* **问题**：需求模糊、进度失控、质量不可测，缺乏标准化流程和经验复用。

### （二）已管理级（Level 2）

* **特征**：建立基本项目管理过程，可管理范围、进度、成本和质量。
* **核心实践**：需求管理、项目计划、监控控制、数据收集，过程可重复但未标准化。

### （三）已定义级（Level 3）

* **特征**：全组织标准化流程，文档化开发、维护和管理过程。
* **核心实践**：过程定义、组织级过程焦点、培训，积累过程资产（模板、最佳实践）。

### （四）量化管理级（Level 4）

* **特征**：通过数据量化管理过程和产品，建立度量体系。
* **核心实践**：量化项目管理、过程性能分析，基于数据预测和决策，提升可预测性。

### （五）优化级（Level 5）

* **特征**：持续改进过程和技术，适应变化，形成创新文化。
* **核心实践**：组织级改进、缺陷预防，快速响应问题，动态优化过程能力。

## 二、图书管理系统项目成熟度评估

### （一）项目背景

这个项目来自于本学期课程《数据库》的期末大作业，我采用 Vue3+Spring Boot 开发图书管理系统，实现基础功能，团队 4人，周期5周。经历需求分析、设计、编码、测试和部署，数据库使用 MySQL。

### （二）成熟度评估维度

#### **1. 需求管理**

* **不足**：
  + 非功能性需求（性能、安全）未明确，需求变更无规范流程，跟踪能力缺失。
  + 案例：新增 "图书预约" 功能未评估，导致进度延误。

#### **2. 项目管理**

* **问题**：
  + 任务划分粗放（如后端接口开发超期 30%），职责不清（代码冲突），风险应对缺失（浏览器兼容性问题未预判）。
  + 工具：仅用 Git 版本控制，进度会议形式化。

#### **3. 过程定义与执行**

* **缺陷**：
  + 无统一编码规范，设计文档不完整（接口变更未更新文档），测试仅停留在功能测试，无用例和性能 / 安全测试。

#### **4. 度量与分析**

* **缺失**：未收集代码行数、缺陷数、开发时间等数据，无质量评估指标（如缺陷率）。

#### **5. 质量保证**

* **薄弱**：代码审查形式化，无独立 QA 团队，缺陷跟踪机制缺失，部分问题遗留到用户阶段。

### （三）成熟度定位

**已管理级初期**：具备基础计划和监控，但流程未标准化，需求、质量、度量管理存在显著短板，未形成组织级过程资产。

## 三、软件过程改进计划

### （一）改进目标

* **短期（1-3 月）**：建立标准化流程和文档模板，完善需求与项目管理。
* **中期（3-6 月）**：引入度量和质量保证，实现过程量化管理。
* **长期（6-12 月）**：形成持续改进文化，向已定义级迈进。

### （二）具体措施

#### 1. 需求管理优化

* 制定《需求规格说明书》模板，明确功能与非功能需求（如并发用户≥50）。
* 建立变更控制流程：变更需经 CCB 评估（影响分析、优先级排序），维护需求跟踪矩阵。

#### 2. 项目管理强化

* 采用 WBS 分解任务，使用 Jira/Trello 跟踪进度，明确成员职责（开发、测试、文档分工）。
* 建立风险登记册，每周评估风险（如技术风险、资源风险），制定应对策略。

#### 3. 过程标准化与工具引入

* **流程规范**：
  + 编码：统一命名规则（如驼峰法），强制注释（关键逻辑、接口说明），使用 SonarQube 扫描代码质量。
  + 设计：完善数据库 ER 图、接口文档（Swagger 自动生成），模板化架构设计文档。
  + 测试：定义单元测试（覆盖率≥70%）、集成测试流程，编写用例模板（输入 / 预期输出 / 执行步骤）。
* **配置管理**：规范 Git 分支策略（主分支 / 开发分支 / 特性分支），禁止直接提交主分支。

#### 4. 度量体系建设

* **关键指标**：
  + 需求变更率、缺陷密度（缺陷数 / 千行代码）、进度偏差率。
* **数据收集**：各阶段结束时填写《度量数据表》，月度分析报告可视化（Grafana 图表）。

#### 5. 质量保证升级

* 设立 QA 角色，制定《质量保证计划》，定期审计过程（如需求评审、代码审查）。
* 使用 Jira 管理缺陷，流程：发现→分类→修复→验证，闭环率需达 100%。
* 强制代码审查（两人以上评审，通过率≥90%），核心模块采用结对编程。

#### 6. 组织级支持

* 建立过程资产库（模板、案例、最佳实践），定期更新共享（如 Confluence 知识库）。
* 季度培训：CMMI 基础、需求工程、测试技术等，团队成员年培训时长≥20 小时。

### （三）实施计划与里程碑

|  |  |  |  |
| --- | --- | --- | --- |
| 阶段 | 时间 | 主要任务 | 里程碑 |
| 启动 | 第 1-2 周 | 成立改进小组，现状评估，制定计划 | 改进计划文档完成 |
| 流程建设 | 第 3-4 周 | 编写需求、设计、测试等流程模板 | 标准化文档发布 |
| 工具部署 | 第 5-6 周 | 部署 Jira、SonarQube，团队培训 | 工具平台可用 |
| 试点实施 | 第 7-10 周 | 小型项目试点新流程，收集反馈 | 试点报告输出 |
| 全面推广 | 第 11-12 周 | 优化流程并全团队推行 | 全员通过流程考核 |

### （四）风险应对

* **抵触风险**：通过试点展示改进效果，开展一对一沟通，强调流程对个人效率的提升。
* **工具学习成本**：提供分步操作手册，安排老带新结对实践，初期允许双轨制过渡。
* **数据缺失风险**：将数据收集纳入项目验收标准，未达标不予结项，设置月度数据检查机制。

## 四、总结

当前项目处于 CMMI 已管理级初期，核心问题在于流程标准化不足、数据驱动缺失和质量保证薄弱。通过建立规范流程、引入度量工具、强化质量控制，逐步提升过程成熟度。改进计划需聚焦短期可落地措施（如模板制定、工具部署），同时规划长期能力建设（如资产库、培训体系）。过程改进是持续迭代的过程，需团队共识与管理层支持，最终实现从依赖个人能力到组织级过程能力的跨越，为后续项目奠定高效、可控的开发基础。