基于 CMMI 模型的 FuturesLab 软件过程成熟度评估与改进方案

一、引言

随着软件项目复杂度的不断提高，过程管理在项目成功中的作用愈发重要。对于高校学生主导的项目而言，采用规范的软件过程管理模型不仅有助于提升项目的开发效率与质量，更能锻炼团队的工程意识和协作能力。本文将基于 CMMI（能力成熟度模型集成）模型，对 FuturesLab 项目进行软件过程成熟度评估，并提出具体的改进方案，以指导项目后续迭代发展。

二、项目简介

在当前高校的课程教学中，由于真实期货交易涉及较高的金融风险与技术门槛，学生很难在真实市场中进行有效的模拟练习，传统的课堂教学难以满足“理论+实践”结合的教学需求。因此，为提升教学质量、增强学生对期货交易核心逻辑的理解，我们小组设计并开发了本项目——**基于 SpringBoot 的 FuturesLab 期货实验室**，作为一套面向教学与实践的模拟交易系统。

该项目通过构建一个简洁、清晰的模拟交易平台，让学生能够在虚拟环境中体验完整的期货交易流程，包括下单、撮合、成交、持仓管理等操作。项目后端基于 SpringBoot 框架开发，具备良好的模块化结构、清晰的业务分层，方便后续维护与功能扩展。前端则采用轻量级的网页界面实现，直观展示行情数据、订单状态与交易结果，方便用户进行交互操作。

本项目力求以贴近真实但简化实现的方式，帮助学生理解期货交易中的核心概念与操作流程，在实践中巩固知识，提升对金融系统设计与开发的综合能力，也为日后进一步学习金融科技和量化交易等方向打下基础。

三、CMMI 模型概述

CMMI（Capability Maturity Model Integration）由卡内基梅隆大学软件工程研究所提出，是用于改进组织软件过程的一种参考框架。CMMI 提供了五个成熟度等级，分别描述了组织在过程管理上的能力和完善程度：

1. 初始级（Level 1）：无序的过程

该阶段的组织缺乏系统的软件过程管理，开发依赖个人能力和应变，项目成功不可预测。

2. 可管理级（Level 2）：可重复的过程

组织建立了项目管理制度，具备基本的计划、需求跟踪、配置管理和质量控制能力，可在相似项目中重复已有经验。

3. 已定义级（Level 3）：已标准化的过程

过程管理已文档化并标准化，建立了组织级的过程定义和知识库，能够支撑跨项目的持续改进。

4. 量化管理级（Level 4）：可度量的过程

组织开始收集和分析过程和产品数据，基于统计和量化手段进行项目控制和质量管理。

5. 优化级（Level 5）：持续优化的过程

组织具备自我优化能力，持续识别过程瓶颈并通过创新和技术手段推动改进。

CMMI 强调过程持续改进，帮助组织通过分阶段推进，不断增强软件开发质量与效率。

四、FuturesLab 项目概况

FuturesLab 是一个以 Java 和 Spring Boot 为技术栈的期货交易模拟系统，涵盖多个模块，包括：

- 实时行情展示
- 模拟交易（市价单/限价单）与撮合
- 用户持仓与风险控制（强平、保证金率计算）
- 成交记录管理
- 论坛模块（用户发帖与评论）

该项目由五人团队共同完成，团队结构如下：

- 1 人：项目经理 & 后端主程序员（负责系统架构与关键模块）
- 1 人：前端开发（Vue）
- 1 人：数据库设计 & 数据分析
- 1 人：测试与质量保障
- 1 人：需求与产品设计

项目采用 GitHub 进行版本管理，日常沟通依托 QQ 群和 PingCode 工具，任务划分和追踪主要依靠 Trello 看板。

五、软件过程成熟度评估

5.1 评估维度

参照 CMMI Level 2 和 Level 3 的核心过程域，对 FuturesLab 项目从以下 6 个维度进行评估：

1. 需求管理（Requirements Management）
2. 项目计划与监控（Project Planning & Monitoring）
3. 配置管理（Configuration Management）
4. 测试与质量保证（Verification & Validation）
5. 软件工程标准与过程文档（Process Definition）
6. 度量与分析（Measurement & Analysis）

5.2 当前成熟度分析

过程域	当前实践	成熟度等级	存在问题
需求管理	以微信群/Word 文档形式记录，版本控制弱	Level 1	缺少正式的需求变更流程，影响开发一致性
项目计划与监控	使用 Trello 记录任务，但无进度分析	Level 2	无燃尽图、估时机制，任务执行依赖成员自觉
配置管理	使用 GitHub 分支管理，存在不规范操作	Level 2	缺乏 PR 审核流程，主分支被频繁覆盖
质量保证	存在基础 JUnit 测试，但未覆盖所有模块	Level 2	缺乏系统测试、手动测试用例文档
过程文档	无正式流程定义，依赖口头协作	Level 1	缺少开发/测试流程文档与规范指南
度量分析	未建立项目数据收集机制	Level 1	不追踪缺陷率、覆盖率等关键指标

5.3 综合判断

FuturesLab 项目目前处于 **CMMI Level 2：可管理级** 初期。虽然已有部分任务管理与配置管理实践，但在需求追踪、质量控制、过程文档与度量体系方面仍不成熟，难以支撑长期演进和跨阶段复用。

六、改进策略与实践计划

6.1 改进目标

未来应以迈向 Level 3 为目标，构建组织级规范与标准化流程。主要目标如下：

- **规范需求管理流程**：每个版本的需求应明确记录、编号与版本控制。
- **强化质量保障**：系统性测试用例、测试报告与缺陷回归制度必须建立。
- **统一过程定义**：明确开发、测试、发布的规范步骤。
- **推进度量体系建设**：收集 Bug 数、提交次数、测试覆盖率等指标。
- **提升协作与可追溯性**：用 Pull Request + Issue 联动方式促进责任明晰。

6.2 改进计划（按角色分工）

周次	改进项	负责人	支持工具
第 1 周	制定并发布《需求管理流程规范》	产品负责人	Typora + GitHub
第 1 周	设立主分支保护规则，强制 PR 合并	技术负责人	GitHub Branch Rules
第 2 周	编写单元测试覆盖报告，按模块补测	测试同学	JUnit + Jacoco
第 2 周	引入代码静态检查工具（Checkstyle）	后端开发	Maven
第 3 周	建立《开发流程规范》《测试流程规范》文档	所有人协作	Markdown + PingCode
第 4 周	使用 Excel 建立基础度量看板：缺陷数、修复周期、PR 数量等	数据分析员	Excel / Sheets

七、预期改进效果

若能按改进计划执行，FuturesLab 项目在以下方面将获得显著提升：

- **一致性提高**：需求与功能匹配更清晰，减少开发返工；
- **质量可控**：模块测试完整，缺陷易于回归定位；
- **协作更高效**：职责明确，文档支撑沟通；
- **可持续发展**：过程标准文档可用于后续项目，积累经验；
- **教学价值提升**：通过真实模拟 CMMI 改进实践，提升团队软件工程认知。

八、结语

FuturesLab 作为一个学生主导的课程项目，尽管面临人力有限、周期短、经验不足等现实问题，但通过引入 CMMI 的思想体系，依然可以在实践中推动过程改进，逐步建立规范的开发流程。这不仅有助于项目自身的稳定运行与交付，也极大增强了团队成员的软件工程意识与实际能力。

通过本次成熟度评估与改进，我们将从“完成项目”走向“管理项目”，实现从混乱到规范、从经验到体系的飞跃，为今后承担更复杂、更正式的项目打下坚实基础。