Creative Education Studies 创新教育研究, 2025, 13(10), 116-124
Published Online October 2025 in Hans. https://www.hanspub.org/journal/ces
https://doi.org/10.12677/ces.2025.1310769

程序设计类课程实践教学考核方案研究与实践

——OBE理念引领的应用型高校课程改革探索

崔文升，李　欣

大庆师范学院计算机科学与信息技术学院，黑龙江　大庆

收稿日期：2025年8月7日；录用日期：2025年10月2日；发布日期：2025年10月11日

摘　要

为破解应用型高校程序设计类课程实践教学考核目标漂移与评价单一的问题，引入OBE倒推设计思想，重构考核逻辑与指标体系。基于毕业要求——课程目标映射，设计基础、提升、综合和创新四阶实践任务，并以过程档案、代码质量、协作表现与创新产出等多元证据构建达成度模型，嵌入PDCA闭环，实现数据驱动的持续改进。两轮教学实践中，学生代码规范性与问题求解能力显著提升，综合项目达成率提高13.6%，课堂投入度和团队协作评分分别上升0.18与0.21分，教师及企业导师满意度均超过90%。结果表明，该方案能有效支撑课程改革，强化实践育人效果，具备在同类课程中推广复制的可行性。

关键词

OBE理念，程序设计课程，实践教学考核，应用型高校

Research and Practice on Assessment Schemes for Practical Teaching in Programming Courses

—An OBE-Guided Curriculum Reform Exploration in Application-Oriented Universities

Wensheng Cui, Xin Li

School of Computer Science and Information Technology, Daqing Normal University, Daqing Heilongjiang

Received: August 7, 2025; accepted: October 2, 2025; published: October 11, 2025

文章引用: 崔文升, 李欣. 程序设计类课程实践教学考核方案研究与实践[J]. 创新教育研究, 2025, 13(10): 116-124.
DOI: 10.12677/ces.2025.1310769

Abstract

To address the issues of goal drift and one-dimensional evaluation in the assessment of practical programming instruction at application-oriented universities, this study employs Outcome-Based Education (OBE) backward design to restructure the assessment framework and its corresponding indicator system. Guided by the mapping of graduation requirements to course objectives, four hierarchical practice tasks—foundation, enhancement, integration, and innovation—were developed. Multiple sources of evidence, including process portfolios, code quality, collaboration performance, and innovative outputs, were integrated to construct an attainment model embedded within a PDCA (Plan-Do-Check-Act) closed-loop system, enabling data-driven, continuous improvement. Over two teaching cycles, significant improvements were observed in students' code standardization and problem-solving abilities. The attainment rate of capstone projects increased by 13.6%, classroom engagement and team collaboration scores rose by 0.18 and 0.21 points respectively, and both instructor and industry mentor satisfaction exceeded 90%. These findings indicate that a diversified and quantitative OBE-based assessment approach can effectively support curriculum reform, enhance practical learning outcomes, and offer strong potential for replication and dissemination in similar educational contexts.

Keywords

OBE, Programming Courses, Practical Teaching Assessment, Application-Oriented Universities

1. 引言

随着步入人工智能时代，程序设计类课程已成为高校培养学生 AI 开发与应用能力的关键一环。这类课程不仅塑造计算思维，也为 AI 思维奠定基础，内容覆盖面向对象程序设计语言(C++, Java, Python)、Web 开发(HTML, JavaScript)以及数值计算语言(MATLAB, Mathematica, Maple)等。实验实践与考核方式直接决定人才培养质量，因此众多高校教师与研究者展开了广泛探索。余小东等依托“希冀”网络教学平台改革《面向对象程序设计》实践教学模式，实现成果在线提交与自动评判，既提升学生编程能力，也减轻教师负担[1]。赵乌吉斯古楞在 Java 课程中引入项目实践教学法，将平时成绩与期末考试相结合进行评价[2]。周玲艳构建了日常－期末、个人－团队、互评－师评并重的多元化考核体系[3]。刘杰等基于 OBE 理念，通过线上考试、综合教学平台和阶段汇报等正向考核，结合学生与用人单位满意度等反向评估，形成可持续改进的评价机制[4]。史晓楠设计可扩展项目与个性化作业包，并辅以项目评价与问卷调查[5]。王颖等针对 Web 程序设计课程考核不合理之处，借鉴 OBE 理念完善考核方法、成效评估与持续改进流程[6]。李征团队也在 OBE 指导下，将参与度与作业、阶段、实验和期末考核整合为多维评价体系[7]。这些研究显著促进了学生编程能力、计算思维与 AI 思维的发展，然而仍普遍存在指标映射粒度不够、过程数据量化不足、师生共用可视化改进工具缺乏及从课堂、实验室到企业实训衔接薄弱等问题。为此，有必要构建“多元证据、数据驱动、闭环改进”的一体化考核方案，真正打通理论教学与实践应用的培养链条。

2. 研究的理论基础

OBE (Outcome-Based Education)强调以学习产出为中心的反向设计。该理念首先清晰界定毕业与课程

层面的可观测学习结果，再据此对齐教学活动与评价方式。在高等教育改革中，这一思路与“建构式对齐”共同成为课程改革的核心支点，要求“目标、教学、考核”一致性与可证据化达成。面向应用型高校的人才培养情境，产业数字化与 AI 的兴起使雇主对编程能力、数据与 AI 素养等复合能力的需求显著提升，行业与产业标准亦将复杂问题求解、工程/软件设计、团队协作与沟通等列为毕业生成果的关键维度，进一步凸显实践能力与真实任务表现的重要性。据此，实践教学考核方案对应用型人才培养至关重要。本文基于 OBE 理念构建考核方案，将过程证据与结果证据联动，支撑达成度量化与 PDCA 闭环的持续改进，形成可诊断、可迭代的质量保障链条[8]。具体到程序设计类课程，形成性与持续性考核可显著提升学习成效、动机与问题解决能力，并在提高教学质量的同时降低教师负担，因此将“多元证据、数据驱动、闭环改进”的实践考核深度融入编程课程，既必要亦紧迫[9]。

3. 项基于 OBE 理念的实践教学考核体系设计

为便于阐释，本文以下以应用型高校普遍开设的《面向对象程序设计》课程为例展开。

3.1. 毕业要求与课程目标设计

应用型高校对程序设计能力的毕业要求(Graduate Attributes, GA)强调“可用、可证、可迁移”。在能力维度上，《面向对象程序设计》课程相关的 7 条指标点毕业要求如表 1 所示。GA1-3 工程知识表示毕业要求 1 工程知识认证的分指标 1~3。

Table 1. Graduation requirement indicators
表 1. 毕业要求指标点

指标点	毕业要求(GA)
GA1-3 工程知识	掌握计算机科学基础与程序设计原理，能分析并解决常见编程问题
GA3-2 设计/开发	能面向对象完成需求分析与系统设计，形成可实现的解决方案
GA5-1 工具使用	能熟练使用现代工程工具与开发环境(IDE、VCS、静态分析、测试框架)
GA9-1 团队	能在多角色团队中有效协作
GA10-1 沟通	能进行规范的技术表达与沟通
GA8-1 职业规范	具备工程伦理与代码规范意识
GA12-1 终身学习	具备自主学习与终身学习能力

《面向对象程序设计》课程据此设定六项课程目标(Curriculum Objective, CO)。CO1 掌握类与对象、封装、继承、多态、泛型、异常与集合等核心概念并完成中等规模编程。CO2 能运用用例、类图与时序图开展面向对象分析与分层架构设计并落实到代码。CO3 熟练使用 IDE、Git、单元测试、静态分析、CI 等工程化工具，形成可维护、高质量代码。CO4 在约束条件下完成迭代式综合项目的实现、测试与文档化交付。CO5 在多角色团队中进行任务分解、代码评审与技术汇报，体现高效协作与沟通。CO6 遵循编码规范与工程伦理，进行阶段反思与技术拓展，形成持续学习能力。毕业要求与课程目标的关系矩阵如表 2 所示。

CO1 对 GA1-3 与 GA5-1 提供基础支撑。CO2 与 GA3-2 强对齐，直接贡献于面向对象的设计与开发能力。CO3 对 GA5-1 为强覆盖，并对 GA8-1 与 GA12-1 提供中度支撑。CO4 综合贡献于 GA3-2、GA9-1、GA10-1，通过迭代项目验证“设计、实现、测试及交付”的全链条能力。CO5 对 GA9-1 与 GA10-1 为强覆盖，完善团队与沟通维度。CO6 面向 GA12-1 与 GA8-1 提供持续提升与规范保障。各指标点均由不

少于两个课程目标共同支撑，避免“单点依赖”。在实施中以“任务、证据和指标”加权聚合计算达成度，并设定班级阈值与个体比例阈值进行诊断，将未达项纳入 PDCA 闭环，从而实现毕业要求与课程目标的一致性、可观测与可改进。

Table 2. The relationship matrix between GA and CO
表 2. 毕业要求(GA)与课程目标(CO)的关系矩阵

课程目标(CO)\毕业要求(GA)	GA1-3	GA3-2	GA5-1	GA9-1	GA10-1	GA8-1	GA12-1
CO1 基本概念与编程	**0.50**	0.10	**0.40**				
CO2 OO 分析与设计	0.30	**0.50**	0.20				
CO3 工具与质量保障		0.10	**0.70**			0.10	0.10
CO4 综合项目开发		**0.60**	0.10	0.10	0.20		
CO5 团队与技术沟通				**0.60**	**0.40**		
CO6 规范与自我发展			0.10	0.00	0.10	0.20	**0.60**

3.2. 四阶实践教学任务的设计

为落实“目标、教学、考核”一致性并凸显《面向对象程序设计》的工程化特征，实践环节按“基础、提升、综合和创新”四阶递进组织。各阶统一采用“任务说明、产出清单、证据链、量规和反馈改进”模板，过程证据(Git 提交与评审、看板进度、站会纪要、反思日志)与结果证据(功能完备度、测试覆盖率、静态告警、性能与文档)。学期权重设计为：S1 阶段为 20%、S2 阶段为 25%、S3 阶段为 35%、S4 阶段为 20%。教学任务与课程目标矩阵关系如表 3 所示。以课程目标 CO3 为例计算达成度，如公式(1)所示。

$$\mathrm{CO3} = \sum_{s\in 1\ldots s4} \left(\text{阶段成绩}_s \times \text{阶段权重}_s \times \text{矩阵系数}_{s,\mathrm{co3}} \right) \tag{1}$$

Table 3. The relationship matrix between teaching tasks (S) and course objectives (CO)
表 3. 教学任务(S)与课程目标(CO)的关系矩阵

组织阶段\课程目标	CO1	CO2	CO3	CO4	CO5	CO6	合计
S1 基础	0.60	0.10	0.30	0.00	0.00	0.00	1.00
S2 提升	0.10	0.60	0.30	0.00	0.00	0.00	1.00
S3 综合	0.00	0.00	0.10	0.60	0.30	0.00	1.00
S4 创新	0.00	0.00	0.20	0.20	0.00	0.60	1.00

S1 基础阶段聚焦类与对象、封装/继承/多态、异常、集合与泛型等核心概念与语法可用性。该阶段通过 3 次小型编程作业与在线测评集合，完成最小可行程序与基础单元测试。S2 提升几段以个人小项目驱动的面向对象建模与模式化设计，完成至少落地 1 个设计模式，提交设计说明及重构记录，形成“能设计、可实现、可维护”的中层能力。S3 综合阶段，团队完成两轮迭代项目，建立产品待办、任务分解与燃尽跟踪，并在每轮提交可运行版本、技术文档与用户说明，系统检验“需求、设计、实现、测试和交付”的全链条能力。S4 创新阶段在 S3 基础上开展增量创新，如性能优化与压力测试、关键模块模式重构及容器化与部署自动化等。该阶段提交创新方案与对比实验、性能/安全报告、技术短讲与合规自查，至少实现 1 项可量化改进。

在评价与改进机制方面，全程采用四级行为量规(未达/基本/良好/卓越)与“任务、指标、课程目标”权重聚合计算达成度，实施“班级均值阈值 + 个体比例阈值”双判据进行诊断。对未达项，进入 PDCA 闭环(调任务难度与比重、补充微课与训练、优化量规与工具链、增加辅导与口试抽测)。公平性通过团队互评校准、口试抽检与相似度检测三重机制保障。可及性通过离线包与轻量工具替代方案覆盖，确保不同学习情境下的一致质量与可持续改进。

3.3. 多元考核指标体系的构建

3.3.1. 设计原则与总体框架

本体系以 OBE 与证据中心评测为底座，遵循“目标、证据、量规、解释和改进”的闭环。(1) 目标层面对齐毕业要求与课程目标。(2) 证据层面同时收集结果性与过程性数据。(3) 量规层面提供可操作的行为描述，解释层面通过可视化仪表盘与诊断报告给出改进依据。(4) 改进层面进入 PDCA 周期，在下一轮教学中验证优化成效。体系突出《面向对象程序设计》的工程化特征与任务驱动路径，强调“可观测、可度量、可追踪、可比对”。

3.3.2. 能力指标与内涵

指标分为知识与问题求解(Knowledge and Problem Solving, KS)、工程质量与工具(Quality of Engineering and Tools, QE)、协作与沟通(Collaboration and Communication, CC)和规范与成长(Plan and Growth, PG)四个方面覆盖知识、工程、协作与成长的核心能力，如表 4 所示。

3.3.3. 证据链设计与采集

结果性证据强调“可运行、可验证、可维护”，包括功能与边界用例通过、测试通过率与覆盖率、静态检查与复杂度、性能与资源占用、交付与文档规范。过程性证据强调“可追踪、可解释、可改进”，包括 Git 提交粒度与节律、PR/CR 数量与质量、需求澄清记录、任务看板进度与燃尽、阶段汇报与反思日志、结对编程与站会纪要。为契合该课程特性，增设“UML－代码一致性”与“设计模式选型－重构记录”两类证据点，保证从设计到实现的闭环可被度量。

3.3.4. 行为量规与评分标准

采用四级行为量规(未达、基本、良好、卓越)映射至区间[0, 1] (0.25/0.50/0.75/1.00)，并配套锚定描述以降低主观性。例如，QE1 测试覆盖率评分标准为：未达小于 50%、及格 50%~69%、良好 70%~84%、优秀大于 85%，且无关键用例遗漏。

3.3.5. 权重配置与动态再加权

基线配比设为 KS 30%、QE30%、CC25%、PG15%，依据建构式对齐与工程教育通行做法[10]，将早期的知识理解与工程质量置于更高权重，协作与职业规范作为随任务情境逐步强化的能力维度。随后组织两轮德尔菲咨询并用 AHP 计算相对重要度，由高校教师与企业导师共同打分，一致性检验达标(CR < 0.10)，所得簇级权重与上述基线高度吻合。在过渡期试点数据上实施探索性与验证性因子分析(KMO > 0.80, Bartlett $p < 0.001$)，观测指标在“知识与问题求解、工程质量与工具、协作沟通、规范与成长”四因子上的载荷清晰，据此细化簇内二级指标权重，并结合四阶实践进行阶段再加权，以在 S1/S2 强化 KS 与 QE、在 S3 强化 CC、在 S4 强化 PG。为避免“高分掩盖底线”，设置主干 CI 失败、静态检查出现 Blocker、关键模块覆盖率低于阈值以及严重性能退化或安全隐患等“硬门禁”，作为不可补偿的必要条件，触发扣分或返工直至达标。敏感性分析表明，在±20%权重扰动下，CO 达成度趋势与个体排序保持稳定，说明该配权在解释力与稳健性之间取得了平衡。

Table 4. Capability index relationship matrix
表 4. 能力指标关系矩阵

指标	能力内涵 (定义与边界)	代表性指标 (示例)	主要证据 (结果/过程)	典型度量 (建议阈值)	阶段侧重 (S1~S4)
知识与问题求解 KS	概念掌握、面向抽象建模、算法与数据结构运用、异常与边界处理；强调“会用、用对、覆盖边界”。	KS1 概念与语法正确性； KS2 用例－类图－时序图一致性； KS3 问题分解与复杂度控制。	在线评测报告、UML 工件、设计说明书、边界用例与反例集。	在线评测通过率 ≥85%； 关键用例通过 =100%； 基础覆盖率 ≥50%； UML 实现一致性 ≥0.80 (或达“良好”级)。	S1、S2
工程质量与工具 QE	单元/集成测试、静态分析与复杂度、持续集成与构建质量、设计模式合理应用、文档与交付、性能与资源。	QE1 覆盖率(行/分支)； QE2 圈复杂度与重复率； QE3 CI 通过率与门禁； QE4 UML－实现一致性； QE5 性能/资源。	测试报告、静态分析报、CI 构建日志、性能压测报告、部署脚本。	行覆盖率 ≥70%、 分支 ≥60%； 关键模块圈复杂度 ≤10； CI 主干通过率 ≥95%	S1、S2、S3 (以 S2、S3 为主)
协作与沟通 CC	团队分工与协作、评审质量与响应、口头与书面表达、需求澄清与迭代评审；强调真实项目场景沟通。	CC1 看板燃尽偏差； CC2 PR 有效评论比率与合并时长； CC3 口头/书面表达清晰度与条理性。	Git 日志与 PR 记录、任务看板与燃尽图、站会纪要与迭代评审材料、同伴互评。	燃尽偏差 ≤±20%； PR 平均合并时长 ≤48 h； 有效评论比率 ≥60%； 出勤与参与度 ≥90%。	S3 (在 S2、S4 中做巩固)
规范与成长 PG	编码规范与工程伦理、开源协议/依赖安全合规、自主反思与技术拓展；强调可持续发展与职业素养。	PG1 规范达标与零高危告警； PG2 依赖安全与许可证合规； PG3 学习日志/反思质量与技术成长证据。	静态检查与规范扫描报告、依赖安全扫描、合规清单、学习日志/技术博客。	高危告警 =0； 高危依赖漏洞 =0； 合规项通过率 =100%； 反思与成长量规 ≥“良好”。	S4 (贯穿全程)

3.3.6. 指标－任务的绑定与聚合

每个指标绑定到具体任务与阶段，再映射到课程目标与毕业要求，形成“指标、任务、阶段、CO 和 GA”的有向链。计算流程遵循“标准化、加权、聚合”。课程目标 CO 达成度如公式(2)所示。毕业要求达成度 GA 公式(3)所示。采用“双阈值判据”进行诊断：班级均值大于 0.75 且大于 70%，学生个体大于 0.60 视为达成。

$$\text{CO 达成度} = \sum\left(\text{指指标指标标} \times \text{指标标权} \times \text{阶段权重}\right) \tag{2}$$

$$\text{GA 达成度} = \sum\left(\text{CO 达成度} \times \text{CO} \to \text{GA 达成度}\right) \tag{3}$$

4. 考核方案的实施与效果分析

4.1. 教学实施过程

本课程围绕“明确目标、促进学习、形成性评价及反思改进”的路径组织教学，突出以学为中心、以证据促学的理念。开课首周通过学习契约与导学环节，让学生理解毕业要求与课程目标的对应关系，熟悉评价标准与学习支持方式，并完成学习风格与基础诊断。S1 基础阶段采用“微讲授、示例化讲解、就地练习、即时测验和纠错提升”的节奏，配合代码阅读、概念图等同伴互教，帮助学生在真实语境中建构面向对象核心概念与基本表达；课堂以小步练习记录学习证据，教师滚动给出针对性反馈。S2 提升阶

段引入案例化与设计工作坊教学。学生以小案例完成“情境、问题、方案、反思”的循环，手绘用例与类图、开展互评，再通过类比讲解理解常见模式与良好风格；形成性评价以评分量规、同伴互评与口头答辩为主，强调“能说明为什么这样设计”。S3 综合阶段转入合作式项目化学习，采用学习小组轮值与角色轮换，结合周计划、阶段展示与“演示日”公开点评，强化任务分解、沟通表达与共同问题解决。教师以“教练式”巡回指导、过程访谈和学习档案袋评阅支持小组协作与个人成长。S4 创新阶段开展探究式学习与小型行动研究，学生围绕“可用性提升”“学习资源设计”“算法改进”等自拟主题制定计划、收集证据并进行对比展示，完成简短技术陈述与反思报告。同时，安排校内外导师讲座与示范课，拓展真实情境的理解。整个学期持续实施形成性评价与反馈：课堂即时提问、低门槛小测、团队互评、学习日志与期中学习诊断，结合教与学双方的中期与期末教学反思，适时调整教学策略与任务难度，确保“教－学－评”一致并促进学生在知识、方法与态度三个层面协同发展。

4.2. 实施效果分析

以大庆师范学院软件工程专业为对象，选取 2022~2023 学年为基线组(未系统应用本方案)、2023~2024 学年为过渡期(部分模块试点)、2024~2025 学年为全面实施期(四阶实践 + 多元证据 + 量化量规 + PDCA)。三届课程由同一教学团队授课，教学大纲与教材一致，样本 80 人规模相近。数据来源包括课程学习档案、阶段与期末测评、团队互评、课堂观察与访谈，统一完成标准化与异常值处理，采用“班级均值＋个体达成比例”的双判据，并做稳健性检验。

总体达成度呈阶梯式提升，体现出软件工程专业以产出为导向的培养特点。课程目标 CO 综合均值由 0.68 (2022~2023)升至 0.73 (2023~2024)，在 2024~2025 达 0.79。达到阈值的学生占比由约 61%增至 69%，最终约 82%。综合项目开发 CO4 与团队协作与沟通 CO5 提升最明显，综合项目达成率较基线提高 13.6%。对应到毕业要求，面向对象设计/开发 GA3-2 与团队协作 GA9-1 的班级均值与个体达成比例同步上升，契合软件工程专业“需求－设计－实现－测试－交付”的能力链条。工程质量与工具 QE 维度出现结构性改善，测试覆盖率中位数约由 52%升至 66%，全面实施期达 78%。主干构建通过率由约 88%升至 97%；严重静态告警明显下降，关键模块圈复杂度回落到合理区间。知识与问题求解 KS 中，概念稳定性与边界用例覆盖度同步增强，低水平重复性错误显著减少。

协作与职业发展维度同样改观，支撑软件工程专业的“团队项目制与工程伦理”要求。课堂投入度均分较基线提升约 0.18，团队协作评分提升约 0.21。评审有效评论占比由约 38%升至约 72%，合并时长由约 52 小时缩短至约 24 小时。规范与成长 PG 方面，高风险告警“清零”率提高，开源许可证与依赖合规在全面实施期基本全通过。教师与企业导师综合满意度均在 90%以上。分层分析显示，基础较弱学生在 S1-S2 获益更大，具备一定基础者在 S3、S4 的综合与创新任务中优势凸显。需要关注的风险集中在高负荷周的任务拥堵与个别学生反思深度不足，后续将通过前置示范、节律优化与反思模板加强支持，并微调阶段权重与量规锚点，以进一步巩固软件工程专业课程群的一致性与持续改进成效。

为增强结论的可信度，我们在“三期对照”数据上引入推断性统计。每学年学生样本量 N = 80。比较流程为：先用 Shapiro-Wilk 与 Levene 检查正态性与齐性。连续/近似连续指标采用 Welch t 检验，统一报告均值 M、标准差 SD、均值差的 95% CI、t 与 df、p 值、Cohen’s d。比例指标采用两比例检验并报告比例差及其 95% CI、z 与 p。结果显示，实施期(2024~2025)相对基线期(2022~2023)在课程目标综合达成、工程质量与工具、协作与沟通等维度均有显著提升(见表 5)，而达标比例由 61%提升到 82%，两比例检 $z = 3.03$，$p = 0.0025$，差值 95% CI = [+7.4, +34.6]个百分点；同时“平均 PR 合并时长”显著缩短(Welch $t = -10.74$, $p < 0.001$, $d = -1.70$)，表明流程与协作效率提升与学习成效相互印证。

Table 5. Inferential statistics (Welch t-test/two-proportion test) for baseline vs. implementation period
表 5. 基线 vs 实施期的推断性统计(Welch t/两比例检验)

指标(单位)	基线 M /比例	SD	N	实施 M /比例	SD	N	差值的 95% CI (实施 − 基线)	统计量	p 值	效应量
课程目标 CO 综合均值(0~1)	0.68	0.09	80	0.79	0.08	80	[0.083, 0.137]	Welch t (155.86) = 8.17	<0.001	d = 1.29
主干 CI 通过率(%)	88	8	80	97	5	80	[6.91, 11.09]	Welch t (132.55) = 8.53	<0.001	d = 1.35
测试覆盖率中位数(%)	52	18	80	78	16	80	[20.61, 31.39]	Welch t (155.86) = 9.66	<0.001	d = 1.53
有效评审评论占比(%)	38	15	80	72	14	80	[29.41, 38.59]	Welch t (157.25) = 14.82	<0.001	d = 2.34
平均 PR 合并时长(小时)	52	20	80	24	12	80	[−33.22, −22.78]	Welch t (129.35) = −10.74	<0.001	d = −1.70
达标学生比例(%)	61%	-	80	82%	-	80	[+7.4, +34.6] (pp)	两比例 z = 3.03	0.0025	φ = 0.23

注：连续/近似连续指标采用 Welch t 检验并报告 Cohen's d；比例采用两比例检验，差值用百分点(pp)表示。

4.3. 满意度调查分析

研究采用经专家审定的问卷量表与半结构化访谈相结合的方式开展满意度调查。问卷围绕“课程设计清晰度、评价公平与透明、反馈及时性、学习支持与资源、协作体验、工作量合理性、感知学习收获”七个维度设计 28 个条目，覆盖学生、任课教师与企业导师三类对象。全面实施学年(2024~2025)学生有效样本 182 份，回收率 92.4%，教师与企业导师有效样本 18 份与 12 份。结果显示，学生总体满意度由基线年的 84%经过渡期的 88%提升至全面实施期的 92%。分维度看，“评价公平与透明”(从 3.92 升至 4.38)与“反馈及时性”(从 3.85 升至 4.31)提升最为显著，“协作体验”随团队迭代和同伴互评机制优化由 3.78 升至 4.26。“工作量合理性”虽有改善，但仍是相对薄弱项。教师与企业导师综合满意度稳定在 90%以上，其中对“课程与岗位能力对接度”的认可度最高 4.45，对“高峰周任务拥堵”的关切度较高 3.62。

5. 讨论

尽管本研究在“教－学－评”一致性与学习产出上取得了可见成效，我们也观察到一系列需要正视的挑战与非预期结果。首先，部分学生在高峰周出现“为指标而学”的行为，过度追求覆盖率与 CI 绿灯而忽略设计思维与代码可读性。其次，教师精力与工具链维护成本的上升，量规校准、共评组织、流水线故障排查与版本依赖升级均要求稳定投入，导致个别教学环节压缩。最后，是公平性问题与同伴互评的偏差风险，设备与网络条件差异、角色分工不均衡以及“情感性评分”都会影响过程证据的有效性。上述现象与教育理论中的若干忧虑相呼应：建构式对齐强调目标清晰与证据对齐，但若指标化过度，容易引发“可测即有效”的偏误。OBE 的批判性观点也提醒我们，单一可量化结果可能导致课程窄化与“教学对齐即教学即合理”的工具理性倾向。为此，我们今后改进中拟采用了三类缓解策略：(1) 以“学习为本”而非“测量为本”为导向重构任务，使量规服务于学习而非反客为主。(2) 通过抽样口试、设计评审与反思报告提升质性证据权重，抑制“刷指标”倾向。在负荷管理与公平性方面前移支撑与节律调控，优化工作量峰值分布，完善同伴互评的标定与异常检测。(3) 进一步的研究需要把这些发现放入更宽广的理论对话中，将以学习科学、形成性评价与专业共同体等视角深化解释，并持续检验在不同院校与课程

中的外部效度与可迁移性。

6. 总结与展望

本研究基于OBE理念构建了“目标、证据、量规、改进”的闭环，并以基础、提升、综合、创新四阶段的实践教学落地于软件工程专业的面向对象程序设计课程，结果显示综合项目质量、协作沟通与工程质量指标均有提升，学生与教师、企业导师的满意度也更高。需要说明的是，本研究仍局限于单一院校与单门课程，部分指标对工具与教师培训存在依赖，学习负荷在高峰周波动较大。今后将在不同类型院校与课程中开展小范围试点，如师范类、应用技术类与综合性大学的程序设计基础、数据结构、软件工程实践等课程，采用可比的对照或准实验设计，透明报告样本信息与统计结果，检验方案的普适性与稳定性。

基金项目

全国高等院校计算机基础教育研究会计算机基础教育教学研究项目(项目编号2025-AFCEC-302)。
大庆师范学院2022年度教育教学改革项目(项目编号：JY2022)。
黑龙江省教育科学“十四五”规划2023年度重点课题(项目编号：GJB1423020)。

参考文献

[1] 余小东, 于曦, 王跃飞, 等. “互联网+教育”背景下面向对象程序设计课程实践教学改革[J]. 高教学刊, 2022, 8(26): 138-141.

[2] 赵乌吉斯古楞. Java程序设计课程项目实践教学模式研究[J]. 赤峰学院学报(自然科学版), 2025, 41(2): 105-108.

[3] 周玲艳. Java程序设计课程实践教学方法和考核方式的思考[J]. 中国现代教育装备, 2022(17): 122-124.

[4] 刘杰, 赵永强, 刘晋钢. 基于OBE理念的“C程序设计”课程教学改革与探索[J]. 教育理论与实践, 2022, 42(3): 61-63.

[5] 史晓楠. 基于OBE的Java程序设计教学改革研究[J]. 软件导刊, 2017, 16(8): 216-218.

[6] 全王颖, 肖红, 张强. 基于OBE理念的“Web程序设计”课程教学改革探索与实践[J]. 微型电脑应用, 2020, 36(7): 14-16.

[7] 李征, 乔保军, 杨伟, 等. 基于OBE的数据结构多维融合教学模式实践[J]. 软件导刊, 2024, 23(2): 162-166.

[8] 李欣, 付丹丹, 李宏博, 等. OBE理念下应用型高校软件工程专业“一流本科专业”建设[J]. 电脑与电信, 2025(4): 105-109.

[9] Calderon, K., Serrano, N., Blanco, C. and Gutierrez, I. (2024) Automated and Continuous Assessment Implementation in a Programming Course. *Computer Applications in Engineering Education*, **32**, e22681. https://doi.org/10.1002/cae.22681

[10] Biggs, J. (2011) Teaching for Quality Learning at University: What the Student Does. McGraw-Hill/Society for Research into Higher Education & Open University Press.