

# 软件压力测试虚拟仿真实验设计与教学实践

张琰彬<sup>1</sup>, 王伟<sup>1</sup>, 黄波<sup>1</sup>, 郭健美<sup>1</sup>, 吴平颐<sup>2</sup>, 朱飞鸽<sup>3</sup>

(1. 华东师范大学数据科学与工程学院; 2. 华东师范大学教师教育实验教学中心, 上海 200062;  
3. 杭州飞致云信息科技有限公司, 浙江 杭州 310024)

**摘要:** 软件性能测试是当今复杂软件研发中不可或缺的环节, 尤其是压力测试对保障软件在用户数量持续增长及大规模并发环境下的稳定性具有决定性作用, 此类测试旨在确保应用系统高效且稳定地运行。然而, 软件压力测试本身面临复杂系统、高成本、长周期和资源投入的挑战, 在有限的教学资源中, 如何有效地传授压力测试的复杂方法 and 专业工具使用技巧, 并激发学生学习兴趣, 以适应不断发展的软件研发生态, 已成为高等教育教学中的难题。为此, 针对软件压力测试实践教学需求, 通过虚拟仿真技术构建了软件性能测试实验平台 VSP4SST, 将真实的3种典型性能故障及其解决策略进行仿真, 帮助学生掌握性能测试的基本工作流程, 确保在有限的教学资源中实验具备可重复、可控和可扩展的特性。实践表明, 该平台已成功应用于本科生和研究生课程, 取得了积极的反馈。

**关键词:** 软件; 性能; 压力测试; 虚拟仿真; 教学

**DOI:** 10.11907/rjdk.241738

扫描二维码阅读全文:



**中图分类号:** G434

**文献标识码:** A

**文章编号:** 1672-7800(2025)011-0153-08

## Virtual Simulation Experiment Design and Teaching Practice for Software Stress Testing

ZHANG Yanbin<sup>1</sup>, WANG Wei<sup>1</sup>, HUANG Bo<sup>1</sup>, GUO Jianmei<sup>1</sup>, WU Pingyi<sup>2</sup>, ZHU Feige<sup>3</sup>

(1. School of Data Science and Engineering, East China Normal University;  
2. Center for Faculty Teaching Development, East China Normal University, Shanghai 200062, China;  
3. Hangzhou Feizhi Cloud Information Technology Co., Ltd., Hangzhou 310024, China)

**Abstract:** Software performance testing is an indispensable part of today's complex software development, especially stress testing, which plays a decisive role in ensuring the stability of software in a continuously growing user base and large-scale concurrent environment. Such testing aims to ensure the efficient and stable operation of application systems. However, software stress testing itself faces challenges of complex systems, high costs, long cycles, and resource investment. In limited teaching resources, how to effectively teach the complex process and professional tool usage skills of stress testing, and stimulate students' learning interest to adapt to the constantly developing software development ecosystem, has become a difficult problem in higher education teaching. To meet the practical teaching needs of software stress testing, a software performance testing experimental platform VSP4SST was constructed through virtual simulation technology. Three typical performance faults and their solution strategies were simulated to help students master the basic workflow of performance testing and ensure that the experiment has repeatable, controllable, and scalable characteristics within limited teaching resources. Practice has shown that the platform has been successfully applied to undergraduate and graduate courses, receiving positive feedback and suggestions.

**Key Words:** software; performance; stress testing; virtual simulation; education

**收稿日期:** 2024-08-26

**基金项目:** 国家自然科学基金项目(62272167)

**作者简介:** 张琰彬(1983-), 女, 华东师范大学数据科学与工程学院工程师、硕士生导师, 研究方向为计算教育学与人工智能垂域智能体; 王伟(1979-), 男, 博士, 华东师范大学数据科学与工程学院教授、博士生导师, 研究方向为开源软件与软件工程; 黄波(1973-), 男, 博士, 华东师范大学数据科学与工程学院教授, 研究方向为软件性能优化; 郭健美(1981-), 男, 博士, 华东师范大学数据科学与工程学院教授, 研究方向为软件性能优化; 吴平颐(1984-), 男, 硕士, 华东师范大学教师教育实验教学中心高级工程师, 研究方向为教育技术; 朱飞鸽(1985-), 女, 硕士, 杭州飞致云信息科技有限公司东区副总经理, 研究方向为软件性能测试产品交付与实施。本文通讯作者: 吴平颐。

## 0 引言

随着人工智能、自动化、区块链、物联网、5G、云计算和大数据的广泛应用,软件研发生态系统日益复杂。随着软件复杂性和用户需求持续增加,大规模并发需求已成为日常工作的常态。在该背景下,确保应用系统的性能和稳定性便显得至关重要<sup>[1]</sup>。软件性能测试在整个软件开发周期中担负着至关重要的责任,其中软件压力测试尤为关键,在保障软件质量和用户体验方面具有至关重要的意义。

本文通过模拟实际用户场景和持续增加负载,通过软件压力测试不断探索待测系统的性能瓶颈,确保系统能在高并发和复杂环境下高效运行,并及早发现、提前解决潜在的性能瓶颈和故障,保证软件上线后的可靠性和稳定性<sup>[2,3]</sup>。压力测试牵涉压测流程、核心技术及性能优化等复杂任务,该领域的工作不仅包括复杂系统测试、多技术堆栈集成、测试环境与真实场景的一致性、负载场景和数据设计等复杂工具和环境,而且成本高、周期长,组织协调困难,需要大量资源支持。

针对企业采用的全链路压测,要求在利用真实生产数据的同时,保证流量隔离、准确预测容量,严格要求生产环境安全,避免潜在的破坏性事件<sup>[4]</sup>。如果要实现规定压测,既要搭建流量平台和改造流量,又要改造数据库、缓存和消息队列等中间件,还需协调多个角色合作,整个团队需要数月时间全力投入,改造数百甚至上千个服务,显示了其复杂性、高成本、长周期,投入的资源较大<sup>[5]</sup>。

为此,本文利用虚拟仿真技术开发了软件性能测试实验平台 VSP4SST(Virtual Simulation Experiment Platform For Software System Stress Testing)。在保持较低资源和费用投入的基础上,整合了自主研发的在线考试系统场景中的 3 个真实性能问题,确保了测试教学过程的可重复性、可控性和可扩展性,旨在深化学生对全链路软件研发流程的理解,认识压力测试在研发过程中的关键作用,掌握压力测试的基本步骤<sup>[6]</sup>。首先,介绍相关研究和挑战,包括软件性能测试教学的相关研究和虚拟仿真技术在教学中的现状;其次,深入介绍虚拟仿真软件性能测试实验平台的架构和设计;再次,介绍虚拟仿真软件性能测试实验平台的计分模型与规则;接下来,阐述目前教学应用场景及用户反馈情况;最后,强调当前面临的问题及未来研究方向。

## 1 相关研究与挑战

### 1.1 性能测试在高校教学中的研究与挑战

近年来,越来越多高校将软件测试纳入计算机科学或软件工程课程,探索如何更好地教授课程或将软件测试融入课程,以帮助学生在实践中理解和应用测试技术。然

而,性能测试实践教学仍存在一些挑战:

(1)复杂性与技术性。性能测试通常涉及复杂的工具和技术,学生具备较高的技术水平才能理解和应用性能测试工具、分析结果及识别性能问题需要,无疑增加了教学难度。如何针对不同学生的技术水平和背景,量身定制教学内容和实践环节,确保所有学生在相对短暂的时间内能获得充分的技能是目前亟需解决的问题<sup>[7-10]</sup>。

(2)测试环境、资源限制。在传授理论知识的同时,如何将性能测试的实际操作融入教学十分关键,由于缺乏实践,学生可能对响应时间、吞吐量和并发用户数等性能指标的含义和重要性感到困惑,导致学生难以正确解释和优化测试结果<sup>[8,9]</sup>。性能测试实践对环境的依赖要求较高,学校可能无法提供充足的访问性能测试工具和实验环境,只能使用免费的软件来构建基础的性能测试教学环境,学生只能通过单机安装软件,在本地进行压力测试。然而,单一的测试工具和本地测试环境难以模拟真实世界的复杂性,无法体验到真实的性能瓶颈和优化流程<sup>[9]</sup>。

部分企业开始提供压力测试服务,通过云服务方式构建真实的待测系统便于学生实践,但真实性能测试环境的构建和维护需要大量资源和技术支持。同时,由于云资源和网络带宽、真实待测系统的接口瓶颈等系统自身的局限性,会导致测试过程不可重复、学生数目受到限制、耗费过高等问题<sup>[10]</sup>。

综上,软件性能测试教学需克服这些挑战,在有限的教学时间和资源上创新教学方法、传授性能测试方法和工具,激发学生学习兴趣和能动性,这也是教学压力测试时需要重点关注的地方。

### 1.2 虚拟仿真在教学中的应用

虚拟仿真教学作为一种创新的教学方法,采用可视化和互动性提供沉浸式学习体验,能模拟微观精密、宏观复杂、贵重或高危环境。在这种教学模式下,学生可在逼真的场景中获取各类知识,包括基本概念、理论和技术,加强对复杂概念的理解。相较于传统实验而言,虚拟仿真教学不仅能避免真实实验环境带来的人身危险和资源浪费,还能为学生提供更灵活的学习环境。

目前,虚拟仿真在农田、医学、建筑、旅游教学等领域的教学中得到了广泛应用<sup>[11-14]</sup>。通过提供完整的模拟环境,使学生参与由教师设计、监督和指导下开发的实践性学习活动,确保过程的可重复性、可控性及成本可控,有效解决了传统教学实践中经费不足、实习资源有限等问题。

本文采用虚拟仿真技术模拟性能测试过程中的性能测试工具及待测系统(System Under Test, SUT)。首先,将实际待测系统中的性能瓶颈问题整合进实验流程;其次,结合数据流可视化、系统架构可视化、压测实施过程可视化等策略,使学生深入理解性能测试的基本流程;最后,为学生提供解决性能问题的具体方案。综上,所提教学方法不仅有助于提升学生的实践操作技能,还有助于深化其对

性能测试领域的认知。

2 基于虚拟仿真的软件性能测试平台架构

2.1 平台技术架构

实验教学平台采用 B/S 架构,分为网络层、展示层、接口层、服务层和基建层,如图 1 所示。

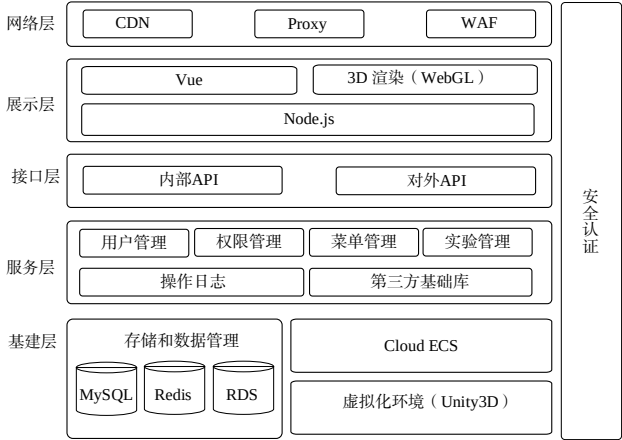


Fig. 1 System technical architecture  
图 1 系统技术架构

整体前端展示层页面基于浏览器,利用 3D Max/Maya、Unity3D 引擎分别进行虚拟场景建模、实验交互内容开发,并将 WebGL 的形式内嵌于浏览器,为教师、学生提供在线虚拟仿真教学模式。服务层管理各模块和前后端及第三方基础库,构建基于云服务的基建层,用户仅需使用浏览器即可进行实验练习,无需安装任何插件或第三方软件。

2.2 平台模块架构

平台分为导学模块、软件压力测试与性能分析实验模块,如图 2 所示。在导学模块中,平台基于背包探险设计了一个开发运维 DevOps 的工作流游戏,使学生回顾理论课程中 DevOps 全链开发过程的每个阶段所需的工具和流程,旨在提升学生对 DevOps 实践的理解,加强其在实际项目中的操作技能<sup>[15]</sup>。软件压力测试与性能分析实验模块包括系统理解与测试过程认知、环境设置与测试脚本设计及压力测试执行与性能分析,通过模拟 3 种典型的性能故障及其解决策略,帮助学生掌握性能测试的基本工作流程,旨在培养学生对软件性能测试的全面认知,使其熟练运用相应工具和方法进行实际应用。

为此,本文通过 VSP4SST 平台,促使学生在虚拟仿真场景深入学习、实践软件性能测试的理论与技术,为未来实际工作作好充分准备。

2.3 课程设计

实验中各模块与相应的理论课程紧密对应,引导模块与 DevOps 软件交付流水线理论课程模块相衔接,共有 4 个学时。课程内容涵盖软件产品整个生命周期的数字化协作流程及其相关工具的使用,详细阐述了产品代码从规

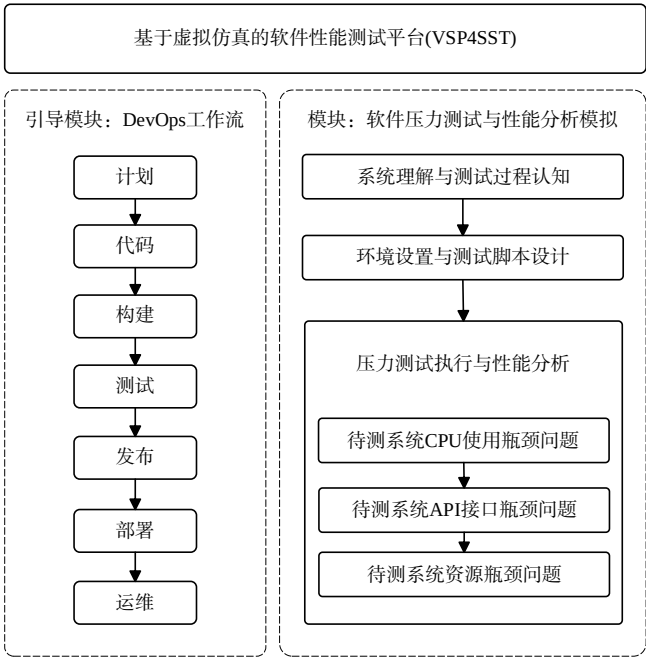


Fig. 2 System module architecture  
图 2 系统模块架构

划、管理至交付的全过程,包括代码提交规范、代码审查、代码构建以及持续集成/持续交付流水线的运作机制,旨在帮助学生深入理解产品从概念到最终交付的基本流程。

此外,软件压力测试与性能分析模拟模块,与 4 个学时的软件性能测试理论课程相互对应,将深入讲解软件性能测试的目标、压力测试的基本步骤及典型案例分析。

2.4 具体模块

2.4.1 引导模块

引导模块旨在复习软件交付流水线理论课程内容,采用游戏化教学策略,将 DevOps 研发流程细分为规划、编码、构建、测试、发布、部署和维护 7 个阶段。实验模块中设计了一个虚拟角色作为游戏主角,引导学生参与实验。实验开始后,系统将自动识别主角当前所处的研发阶段,虚拟界面左右两排小球代表不同工具或角色名称,每个小球上附有文字说明。

为了成功推进至下一阶段,学生须从这些小球中挑选出恰当的工具或操作步骤,将其添加至知识背包。例如,在构建阶段的选项包括: Pull Request、Build、Code Review、Git Commit、Testing、Release、Monitor 及 Maven、Gradle、MsBuild、Selenium、LogStash、CFEngine 等工具,学生应选择与构建阶段相关的 Pull Request、Build、Code Review、Git Commit 及 Maven、Gradle、MsBuild 等工具加入背包,并进行后续实验。

当所有阶段任务完成时,平台将呈现一个详尽的 DevOps 全链路研发流水线图解,以辅助学生全面掌握 DevOps 流程。图 3 为引导模块的运行界面截图及最终的 DevOps 流程回顾全景图。

2.4.2 软件压力测试与性能优化实验模块

该模块旨在通过性能测试模拟实验,使学生熟练掌握



Fig. 3 Guidance module

图3 引导模块

性能测试的基本流程和相关问题解决方案。与 MeterSphere 开源项目 (<https://github.com/metersphere>) 专业测试团队讨论沟通之后,将实验流程划分为业务理解与流程认知、环境设置与脚本设计、压测实施与分析3个关键模块,引入真实产品的接口文档、测试用例及3个实际性能故障案例,帮助学生全面理解性能测试流程。

(1)业务理解与流程认知。性能工程是一个广泛的概念,综合考量整个系统性能的核心在于理解整个系统架构,包括硬件、软件、配置、性能属性及他们之间的相互关系。因此,实验平台首先应呈现系统的全景架构图,包括性能测试工具架构和待测系统的架构,如图4所示。其中,性能测试工具的架构参考了以 MeterSphere 为代表的性能集成工具,待测系统以华东师范大学数据科学与工程学院自研的在线学习系统的考试模块为原型<sup>[6]</sup>。

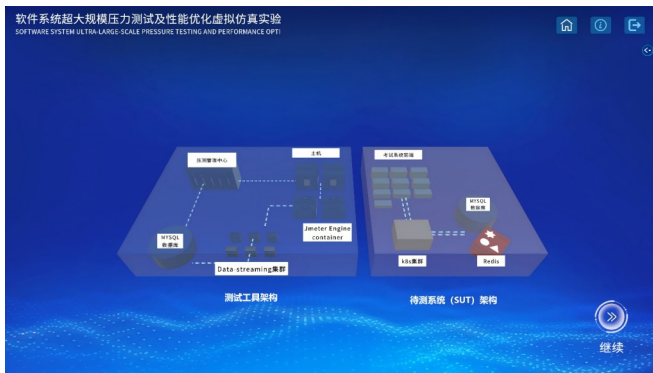


Fig. 4 System panoramic architecture

图4 系统全景架构

为了提升实验场景的真实性,模块纳入了真实产品的接口文档、测试用例及3个具体的实际性能故障案例。这些数据来自华东师范大学数据科学与工程学院自研的在线学习系统的考试模块的真实性能测试数据,并经过整理和分析后融入实验平台<sup>[6]</sup>。学生通过这些真实案例数据,更能直观掌握性能测试流程,学习如何分析和解决实际问题。

此外,通过构建可视化模块展示性能测试工具的架构,及考试系统前后端模块的待测系统架构图。图4虚线表示压测数据流在系统中的动态传递过程,基础水平的学生也能清晰地追踪、理解压力测试数据流动的路径。

(2)环境设置与脚本设计。环境设置旨在模拟真实的压测工具和监控工具的安裝过程。学生在虚拟的电脑桌面上模拟安装待测工具,包括压测工具 JMeter、监控工具 Prometheus 等,先让学生熟悉压测工具和监控工具的安裝过程,为后续实验提供基础。在 JMeter 脚本设计阶段,实验平台提供待测系统真实的接口文档并构建测试需求。学生通过查看测试需求,利用提供的样例脚本,在虚拟仿真系统中导入 JMeter 样例脚本,并修改样例脚本内容来理解脚本设计原则。

图5为系统的脚本设计界面,全面呈现了 HTTP Request 的参数信息,需准确填写端口、服务器端口或 IP 地址等关键信息,以完成接口的压力测试配置。针对基础较薄弱的学生,实验界面右侧提供了针对性的提示信息,学生将通过实际操作理解脚本的设计原则,以提升实际应用能力,为实施性能测试奠定基础。

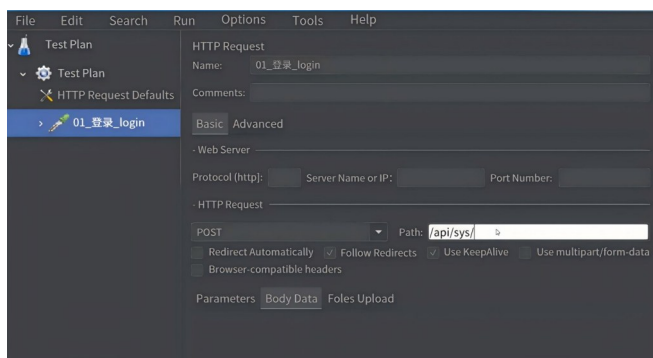


Fig. 5 Script design page

图5 脚本设计页面

(3)压测实施与分析。压力测试主要用于检测待测系统的性能瓶颈点,但在实际压力测试过程中需要软件测试团队与研发团队合作。测试步骤是逐步增加系统负载,实时监控、观察待测系统的性能变化情况,确定在何种负载条件下使系统性能处于失效状态,并在分析相应的性能问题后进行优化和改进,这也是一个持续探索软件性能瓶颈的过程。

为了使学生真正理解该过程,平台在真实的待测考试系统性能测试中,选取待压系统的 CPU 瓶颈、接口瓶颈、服务器容量瓶颈这3个真实性能问题进行模拟,作为实验检查点。目标是通过构建可复现的性能瓶颈,让学生理解整个测试流程及解决相关问题的方法。

具体地,学生登录虚拟压力测试工具页面进行压测配置,测试过程中逐渐提升并发用户数,利用压力测试报告、待测系统监控数据、接口报文等信息揭示系统性能瓶颈。如图6所示,为了深化学生理解压测实施的关键要素和流程,虚拟仿真系统提供了交互性压测流程模拟实验,学生可在压力测试实施配置界面选择关键参数,在页面输入并发用户数,点击保存并执行按钮,系统将自动进行压测,页面右上角会实时反馈实验用时、时间进度和当前实验得分情况。此外,页面右侧的提示符号会给出相应的提示信

息,帮助学生更顺利地进行实验操作。



Fig. 6 Performance pressure test configuration page

图 6 性能压测配置页面

本文通过设置不同的并发数用户数,自动激活预设的 3 个性能瓶颈场景,进而指导学生分析、解决问题。图 7 为系统激活 CPU 瓶颈问题场景时的几个关键步骤。

首先,当压测触发 CPU 性能瓶颈问题时,系统会提供



Fig. 7 CPU performance bottleneck experiment

图 7 CPU 性能瓶颈实验

预设的真实测试报告,学生通过查看报告可看到整体系统仅为吞吐下载过快,未发现其他错误信息;其次,系统引导学生查看待测系统(SUT)的 CPU 利用率报告,揭示 CPU 利用率过高,几乎达到满负荷状态;再次,为了帮助学生更全面地理解该场景,实验提供了系统状态的可视化展示,从架构视角、CPU 视角、服务器视角将真实场景中的关键指标具象化;最后,配合相关的交互式问答来辅助学生掌握整个场景全流程,进而深入了解 CPU 性能评估问题的关键流程和指标。

3 实验平台计分模型与规则

3.1 操作成绩评估模型与计分规则

本文操作成绩评估模型主要针对用户对实验平台的操作技能进行评估,综合考虑了用户完成操作步骤的用时、挑战次数和步骤完成情况,如表 1 所示。

(1)用时评分。模型要求用户在规定时间内完成操作步骤,根据完成情况进行评分。例如,在 DevOps 产线配置模块中每个环节都设置了 60 s 的合理用时限,用户在规定时间内完成操作才能获得满分。

(2)挑战次数评分。模型允许用户尝试多次,但分数会随着尝试次数增加而减少。例如,1 次完成得满分,2 次得 1 分,3 次及以上不得分。这种设计可激励用户在第一次尝试中尽力完成操作,避免过度依赖尝试次数来解决问题。

(3)步骤完成情况评分。模型根据用户操作进行评分,例如填写空格、选择正确答案等,以进一步细化评估。这种设计可确保用户完成各项操作步骤、理解操作的意义和目的。

Table 1 Scoring model and operational scoring rules

表 1 计分模型与操作得分

| 步骤名称                    | 目标要求                     | 合理<br>用时/s | 赋分模型                                                                                         | 满分 | 对应模块<br>完成度/% | 实验完成度<br>(实验进度)/% | 对应<br>实验模块 |
|-------------------------|--------------------------|------------|----------------------------------------------------------------------------------------------|----|---------------|-------------------|------------|
| Devops 生产线配置<br>—规划环节操作 | 掌握规划环节的<br>工作任务和工作<br>工具 | 60         | 根据步骤用时和挑战次数计算分数,步骤用时 60s 内得满分 1 分,超过不得分,计算公式=用时得分+次数得分,1 次完成得满分 2 分,2 次得 1 分,3 次及以上不得分直接下一环节 | 3  | 15            | 5                 | —          |
| DevOps 生产线配置<br>—代码环节操作 | 掌握代码环节的<br>工作任务和工作<br>工具 | 60         | 根据步骤用时和挑战次数计算分数,步骤用时 60s 内得满分 1 分,超过不得分,计算公式=用时得分+次数得分,1 次完成得满分 2 分,2 次得 1 分,3 次及以上不得分直接下一环节 | 3  | 30            | 10                | —          |
| DevOps 生产线配置<br>—构建环节操作 | 掌握构建环节的<br>工作任务和工作<br>工具 | 60         | 根据步骤用时和挑战次数计算分数,步骤用时 60s 内得满分 1 分,超过不得分,计算公式=用时得分+次数得分,1 次完成得满分 2 分,2 次得 1 分,3 次及以上不得分直接下一环节 | 3  | 45            | 15                | —          |
| DevOps 生产线配置<br>—测试环节操作 | 掌握测试环节的<br>工作任务和工作<br>工具 | 60         | 根据步骤用时和挑战次数计算分数,步骤用时 60s 内得满分 1 分,超过不得分,计算公式=用时得分+次数得分,1 次完成得满分 2 分,2 次得 1 分,3 次及以上不得分直接下一环节 | 3  | 60            | 20                | —          |
| DevOps 生产线配置<br>—发布环节操作 | 掌握发布环节的<br>工作任务和工作<br>工具 | 60         | 根据步骤用时和挑战次数计算分数,步骤用时 60s 内得满分 2 分,超过不得分,计算公式=用时得分+次数得分,1 次完成得满分 2 分,2 次得 1 分,3 次及以上不得分直接下一环节 | 4  | 75            | 25                | —          |

| 续表                      |                          |            |                                                                                              |    |               |                   |            |
|-------------------------|--------------------------|------------|----------------------------------------------------------------------------------------------|----|---------------|-------------------|------------|
| 步骤名称                    | 目标要求                     | 合理<br>用时/s | 赋分模型                                                                                         | 满分 | 对应模块<br>完成度/% | 实验完成度<br>(实验进度)/% | 对应<br>实验模块 |
| DevOps 生产线配置<br>—部署环节操作 | 掌握部署环节的<br>工作任务和工作<br>工具 | 60         | 根据步骤用时和挑战次数计算分数,步骤用时 60s 内得满分 2 分,超过不得分,计算公式=用时得分+次数得分,1 次完成得满分 2 分,2 次得 1 分,3 次及以上不得分直接下一环节 | 4  | 90            | 30                | —          |
| DevOps 产线配置<br>—维护环节操作  | 掌握维护环节的<br>工作任务和工作<br>工具 | 60         | 根据步骤用时和挑战次数计算分数,步骤用时 60s 内得满分 2 分,超过不得分,计算公式=用时得分+次数得分,1 次完成得满分 2 分,2 次得 1 分,3 次及以上不得分直接下一环节 | 4  | 100           | 35                | —          |
| prometheus 安装           | 完成 prometheus<br>安装      |            | 根据步骤完成情况计分:<br>1. 完成 docker 安装脚本,2 个空,共计 4 分,一次完成 4 分,2 次完成 2 分,3 次及以上完成 0 分                 | 4  | 10            | 37                | 二(1)       |
|                         |                          | 100        | 2. 完成 prometheus 容器健康查看,1 个空,一次完成 4 分,2 次完成 2 分,3 次及以上完成 0 分                                 | 4  | 20            | 39                |            |
|                         |                          | 100        | 3. 根据结果,完成 prometheus 容器是否健康的判读,正确 2 分,错误 0 分                                                | 2  | 30            | 40                |            |
| prometheus 验证           | 完成 prometheus<br>安装      | 180        | 根据操作情况得分:一次完成 4 分,2 次完成 2 分,3 次及以上完成 0 分                                                     | 4  | 100           | 50                | 二(1)       |
| 软件压测架构<br>认知            | 完成软件压测架<br>构认知           | 60         | 完成软件架构学习得满分                                                                                  | 3  | 20            | 55                | 二(2)       |
| Jmeter 压测脚本<br>设计       | 完成 jmeter 压测<br>脚本设计     |            | 根据步骤完成情况得分:<br>1. 完成线程组新建,2 分                                                                | 2  | 30            | 56                | 二(2)       |
|                         |                          | 100        | 2. 添加 http request defaults,2 分                                                              | 2  | 40            | 56                |            |
|                         |                          | 100        | 3. 根据 IP 和 port 填写完成情况得分,2 个空,1 次完成 5 分,2 次完成 2 分,3 次及以上不得分,且自动填充正确答案                        | 5  | 60            | 57                |            |
|                         |                          | 100        | 4. 添加 http request,2 分                                                                       | 2  | 70            | 57                |            |
|                         |                          | 100        | 5. 查看接口文档,2 分                                                                                | 2  | 75            | 58                |            |
|                         |                          | 100        | 6. 填写接口地址,1 次完成 5 分,2 次完成 2 分,3 次及以上不得分                                                      | 5  | 95            | 59                |            |
|                         |                          | 100        | 7. 另存 jmx 文件,2 分                                                                             | 2  | 100           | 60                |            |
| 压测脚本加载                  | 完成压测脚本加<br>载             | 180        | 加载 jmx 压测文件得 2 分;IP 地址连续 3 次输错,此步骤 0 分                                                       | 2  | 20            | 62                | 二(3)       |
| 压测资源池设置                 | 完成压测资源池<br>设置完成          | 60         | 设置完成得 2 分                                                                                    | 2  | 30            | 64                | 二(3)       |
| 添加压测监控                  | 完成压测监控添<br>加             | 60         | 添加完成得 5 分                                                                                    | 5  | 35            | 65                | 二(3)       |
| CPU 问题测试及性能<br>优化       | 发现 CPU 问题并<br>优化         |            | 成功测出 cpu 问题并解决可以得分,2 道选择题+视角查看+操作分:                                                          | 3  | 40            | 70                | 二(3)       |
|                         |                          | 150        | 1. 压测过程中查看了 3 个视角,3 分;看一个得 1 分                                                               | 1  | 45            | 72                |            |
|                         |                          | 150        | 2. 异常问题分析,2 道选择题,每题 1 分,总共 2 分,答错 3 次及以上不得分,并显示正确答案                                          | 1  | 50            | 75                |            |
|                         |                          |            | 3. 按照后续操作,完成优化,5 分                                                                           |    |               |                   |            |

| 续表               |                |            |                                                                       |    |               |                   |            |
|------------------|----------------|------------|-----------------------------------------------------------------------|----|---------------|-------------------|------------|
| 步骤名称             | 目标要求           | 合理<br>用时/s | 赋分模型                                                                  | 满分 | 对应模块<br>完成度/% | 实验完成度<br>(实验进度)/% | 对应<br>实验模块 |
| 接口问题测试<br>及性能优化  | 发现接口问题并<br>优化  | 150        | 成功测出接口问题并解决可以得分,2道选择题+视角查看+<br>操作分:                                   | 3  | 60            | 83                | 二(3)       |
|                  |                | 150        | 1. 压测过程中查看了3个视角,3分;看一个得1分                                             | 1  | 65            | 84                |            |
|                  |                | 150        | 2. 异常问题分析,2道选择题,每题1分,总共2分,答错3次<br>及以上不得分,并显示正确答案<br>3. 按照后续操作,完成优化,5分 | 1  | 70            | 85                |            |
| 服务器问题测试<br>及性能优化 | 发现服务器问题<br>并优化 | 150        | 成功测出接口问题并解决可以得分,2道选择题+视角查看+<br>操作分:                                   | 3  | 85            | 95                | 二(3)       |
|                  |                | 150        | 1. 压测过程中查看了3个视角,3分;看一个得1分                                             | 7  | 100           | 100               |            |
|                  |                |            | 2. 异常问题分析,服务器扩容,总共7分,选择第一个选项7<br>分,第二个5分,第三个3分,第四个1分                  |    |               |                   |            |

4 教学与学习体验实验应用与反馈

目前,实验课程已成功纳入两门课程,为华东师范大学数据科学与大数据技术专业(简称数据专业)的基础课程及研究生专业课程提供服务,具体为研究生小班课程软件开发管理与实践和本科生课程云计算系统,后者已被认定为国家首批国家级一流本科专业建设点。

该实验平台自校内部署以来已运行4轮,在本科生、研究生教学中已实现多轮次的有效应用,累计实验浏览量达到3 200次,参与实验的学生人数为330人。在教育教学改革过程中,课程反馈是评估、优化教学效果的重要环节。本文通过分析研究生和本科生的课程反馈情况,探讨了实验课程对学生理解研发流程和测试流程的帮助程度、实验难度感知及学生对未来课程开发的兴趣。

研究生的课程反馈显示,所有学生(100%)都认为实验能帮助自身理解整个研发流程和测试流程。在实验难度方面,10%的学生认为非常难,20%的学生认为较难,70%的学生认为难度适中。对于未来围绕该实验开发的课程方面,50%的学生表示非常感兴趣,50%的学生表示较感兴趣。

本科生的课程反馈显示,75.68%的学生认为实验能帮助自身理解研发流程和测试流程。在实验难度方面,21.62%的学生认为非常难,45.95%的学生认为较难,32.43%的学生认为难度适中。对于未来课程开发方面,16.22%的学生表示非常感兴趣,59.46%的学生表示较感兴趣,13.51%的学生表示无所谓,10.81%的学生表示不感兴趣。

同时,学生还对实验模块提出了具体建议。对于实验模块一,建议增加实验解释和步骤讲解、配置前置理论课程、改进用户界面、增加趣味性问答等功能。对于实验模块二,建议着重增加趣味性交互问题和更多案例,为学生通过更多操作空间。

此外,本文针对实验结果的偏差进行了深入分析。在研究生课程中因全面介绍了研发流程、性能测试,并进行实验环节,学生能深入理解概念、提升实际操作能力。在本科生课程中,学生在讲解完云原生研发交付流程后直接参与实验,因此部分学生认为实验难度较高,这一分析有助于改进课程设计,以便更好地适应不同层次学生的学习需求。

5 结语

本文通过虚拟仿真技术构建了软件性能测试实验平台VSP4SST,为软件性能测试教学提供了一种创新的实践教学方法。该平台确保了测试实验教学过程的可重复性、可控性和可扩展性,学生能全面理解全链路软件研发流程,深入认识压力测试在研发过程中的作用及其基本步骤。

本文旨在不断迭代和完善VSP4SST平台,将其建设为软件性能测试领域的教学标杆,为培养具有实战能力的软件工程人才贡献力量。同时,持续关注学生反馈和行业发展趋势,不断优化教学策略,提升实验平台的教学效果,以期软件工程教育领域发展提供参考与借鉴。

目前,平台处于初期发展阶段,但其潜力已初步显现。未来,将深化实验内容与形式的多样化,以更好地满足不同学生的学习需求,提升实验互动性和趣味性;拓展跨专业软件应用的教学内容,以适应多学科背景下的软件研发教育需求,促进技术在不同领域的融合与创新;加强系统的稳定性与健壮性,确保平台适应不断增长的用户群体和复杂的教学场景。

参考文献:

[1] SÖREN H, HASSELBRING W. A configurable method for benchmarking scalability of cloud-native applications[J]. Empirical Software Engineering, 2022, 27(6):143.

[2] PRATAMA M R, KUSUMO D S. Implementation of continuous integra-

- tion and continuous delivery (CI/CD) on automatic performance testing [C]// 9th International Conference on Information and Communication Technology, 2021: 230–235.
- [3] PARAGANKAR S. A comprehensive review of performance testing methodologies and best practices: software quality engineering[J]. International Journal of Science and Research, 2023, 12(8): 2008–2014.
- [4] JIANG J, LI W H, ZHANG L, et al. ShadowDB: a SQL engine based traffic-split system for full-link stress test[J]. Computer Engineering and Science, 2023, 45(12): 2165–2174.  
姜俊, 李文慧, 张亮, 等. ShadowDB: 基于 SQL 引擎的全链路压力测试分流系统[J]. 计算机工程与科学, 2023, 45(12): 2165–2174.
- [5] QIU K, ZHENG Z, TRIVEDI K S, et al. Stress testing with influencing factors to accelerate data race software failures[J]. IEEE Transactions on Reliability, 2019, 69(1): 3–21.
- [6] WANG W, LU X, HUANG B. Shuishan online: constructing and teaching with a data-driven learning platform [C]// IEEE International Conference on Engineering, Technology and Education, 2021: 1–8.
- [7] GAROUDI V, RAINER A, LAUVÁS P J, et al. Software-testing education: a systematic literature mapping [J]. Journal of Systems and Software, 2020, 165: 110570.
- [8] LEMOS O A L, SILVEIRA F F, FERRARI F C, et al. The impact of software testing education on code reliability: an empirical assessment [J]. Journal of Systems and Software, 2018, 137: 497–511.
- [9] ZHAO W D, WANG S Q, ZHANG F. Experimental teaching reform of software load and pressure testing based on LoadRunner [J]. Computer Education, 2019(11): 104–107.  
赵卫东, 王树强, 张峰. 基于 LoadRunner 的软件负载与压力测试实验教学改革[J]. 计算机教育, 2019(11): 104–107.
- [10] DONG L, XIONG J, YANG C H. Study of performance testing of information system based on domestic CPU and OS [C]// Third International Conference on Trustworthy Systems and their Applications, 2016: 112–116.
- [11] CHENG C, CHEN Z J, GUO Z M, et al. Research and development of simulation training platform for multi-agent collaborative decision-making [J]. Journal of System Simulation, 2023, 35(12): 2669–2679.  
成城, 陈智杰, 郭子铭, 等. 多智能体协同决策仿真平台研究与开发 [J]. 系统仿真学报, 2023, 35(12): 2669–2679.
- [12] ZHANG L L, DENG Q X, BAO Y B, et al. Research on virtual simulation experiment of intelligent agriculture based on Internet of Things technology [J]. Research and Exploration in Laboratory, 2022, 41(11): 200–203.  
张立立, 邓庆绪, 鲍玉斌, 等. 基于物联网技术的智慧农业虚拟仿真实验研究设计与应用 [J]. 实验室研究与探索, 2022, 41(11): 200–203.
- [13] CHAI N, PENG R, XU Z L, et al. Application of medical image virtual simulation system in clinical teaching [J]. Medical Education Research and Practice, 2022, 30(1): 108–114.  
柴娜, 彭锐, 徐子良, 等. 医学影像虚拟仿真系统在临床教学中的应用 [J]. 医学教育研究与实践, 2022, 30(1): 108–114.
- [14] YAN L, WEI Z, ZHU Z H, et al. Design and application of virtual simulation experimental teaching system for building fire protection engineering [J]. Fire Science and Technology, 2023, 42(7): 1018–1022.  
颜龙, 魏峥, 朱志辉, 等. 建筑工程消防虚拟仿真实验教学系统设计与应用 [J]. 消防科学与技术, 2023, 42(7): 1018–1022.
- [15] JABBARI R, ALI N B, PETERSEN K, et al. What is DevOps? a systematic mapping study on definitions and practices [C]// Proceedings of the scientific workshop proceedings of XP2016, 2016: 1–11.  
(责任编辑: 刘嘉文)