敏捷开发与协同工作的技术支持：互联网软件开发工具和平台的评估与比较

张春栋

Hoyoverse　Singapore

【摘要】该文对支持敏捷开发，协同工作等互联网软件开发工具及平台进行了全面评价。首先讨论敏捷开发工具与平台是怎样通过它们的协同功能来支持团队合作的。在评价热门敏捷工具协同功能的基础上，分析协同工作是怎样优化敏捷开发流程，提高团队效能的。然后，本文对主要版本控制系统，持续集成/部署工具，云服务平台以及集成开发环境与代码编辑器进行对比分析，从而揭示其功能特点，性能稳定性，成本效益和用户体验。最后根据实际应用场景提供工具及平台选择建议，目的是为软件开发团队提供决策支持。

【关键词】敏捷开发；协同工作；版本控制系统；持续集成/持续部署

【收稿日期】2024 年 5 月 1 日　【出刊日期】2024 年 6 月 12 日　【DOI】10.12208/j.aics.20240007

Technical support for Agile development and collaborative work: Evaluation and comparison of Internet software development tools and platforms

Chundong Zhang

Hoyoverse　Singapore

【**Abstract**】This paper gives a comprehensive evaluation of Internet software development tools and platforms that support agile development and collaborative work. We begin by discussing how agile development tools and platforms support teamwork through their collaborative capabilities. On the basis of evaluating the collaborative function of popular agile tools, this paper analyzes how collaborative work can optimize agile development process and improve team effectiveness. Then, this paper compares major version control systems, continuous integration/deployment tools, cloud service platforms, and integrated development environments with code editors to reveal their functional characteristics, performance stability, cost effectiveness, and user experience. Finally, suggestions on tool and platform selection are provided according to actual application scenarios, with the purpose of providing decision support for software development teams.

【**Keywords**】Agile development; Work together; Version control system; Continuous integration/continuous deployment

1 敏捷开发与协同工作的技术支持

1.1 敏捷开发工具和平台对协同工作的支持

1.1.1 敏捷开发工具的协同功能评估

敏捷开发工具对推动团队成员间的合作发挥着关键作用。这类工具一般提供任务管理，敏捷看板，迭代跟踪以及实时沟通，而这正是协同工作得以实现的基石所在。比如，利用类似 JIRA 等工具，小组可以在让全体成员立即看到项目进度及变化的情况下，很容易创建并追踪用户故事，任务及 bug 等。另一方面，JIRA 提供了一种视觉化的方式来组织工作流，通过卡片和看板来管理项目的各个方面，增强了团队成员之间的透明度和沟通。

1.1.2 敏捷开发平台的协同功能评估

敏捷开发平台把敏捷开发工具功能聚集在更综合的环境下，提供团队协作集成化解决方案。例如 Atlassian 公司的产品 Confluence 和 JIRA 等这些平台不但支持敏捷开发做法，而且更深入知识共享、团队协作等。它们为团队提供了创建共享文档库的能力，允许他们进行实时的合作编辑，确保了信息的流畅传递和知识的持续传递。敏捷开发平台使得从项目规划到代码部署的每一个步骤都在同一环境下完成，这极大地简化了工作流程并减少了上下文切换的成本。

1.2 协同工作对敏捷开发的影响和作用

1.2.1 协同工作对敏捷开发流程的优化

有效的合作可以保证信息快速流转，减少错误认识与拖延，使需求了解，功能发挥，解决问题更快，更准。举例而言，开发人员，测试人员与产品经理通过同一个平台进行实时的讨论，就可以迅速形成一致的意见，从而保证了对需求的正确理解与执行。另外协同工作增强团队成员间的信任感与责任感，每一个成员都为自己的共同目标负责，这种团队精神有利于促进项目前进。敏捷流程的不断反馈循环还取决于好的协同工作。团队成员要能自由提供并接收反馈。这种公开的沟通文化对持续改进非常关键。

1.2.2 协同工作对团队效能的提升

团队效能在敏捷开发中至关重要，而协同工作正是推动这一点的关键因素。当团队成员建立起有效协作机制后，人与人之间互补的技能就会发挥出来，从而推动团队形成合力，共同应对复杂难题。举例而言，开发者能够一边进行编码一边与设计师进行实时交流，以保证所执行的各项功能不仅达到技术要求，而且达到用户体验标准。这种跨职能协作加速了决策过程和产品质量。在合作工作的背景下，团队成员共同追求目标和承担责任，团队的成功与每个成员的成功都是紧密相连的，这种双赢的思维方式鼓励团队成员不断提高自己的表现。

2 互联网软件开发工具和平台的评估

2.1 版本控制系统的比较（如 Git, SVN）

Git 和 SVN 作为目前应用较多的两个版本控制系统各有其特点及优点。Git 是一个分布式版本控制系统，每个开发者都保有代码库的完整副本，这使得操作高效且不依赖于中央服务器。Git 这一设计支持高并行开发，开发者可在当地提交修改，然后同步到中央仓库。相比之下，SVN 是一个集中式版本控制系统，所有的更改都保存在中央服务器上。尽管这样使管理与控制权限变得更简单、更直接，但是会带来网络延迟、单点故障等一系列问题。Git 在分支与合并方面提供很强的功能，对敏捷、更迭频繁的工程尤为实用。而且 SVN 对大型文件以及二进制文件都有较好的处理性能，历史版本比 Git 更容易获得更直接。

2.2 持续集成/持续部署工具(如 Jenkins, Travis CI）

持续集成/持续部署（CI/CD）工具在现代软件开发过程中起到了不可替代的作用，它们为软件开发的自动化流程提供了支持。Jenkins 作为开源自动化服务器，可配置海量插件以支持建设，部署和自动化测试多种工作。其可扩展性与灵活性使其成为众多公司的优先选择，尤其适用于要求高定制 CI/CD 流程。相比而言，Travis CI 所提供的用户体验更简洁，其直接整合在 GitHub 上并支持云端构建使用户能够快速发起 CI/CD 过程。这一即开即用的功能，对那些想要降低配置与维护工作量的队伍是十分诱人的。

2.3 云服务平台（如 AWS, Azure, 阿里云）

云服务平台提供灵活，可扩展的软件开发与部署基础设施。AWS 被认为是市场上最为成熟的云服务供应商之一，它提供了一系列广泛的服务和工具，这些服务和工具覆盖了计算、存储、数据库和网络等多个领域。其全球数据中心网络保证高可用性、低延迟，对那些要求全球覆盖的应用程序尤为重要。在微软云平台 Azure 和 AWS 之间存在着激烈的竞争，它们提供紧密整合的业务，特别是对依靠微软技术栈进行业务。Azure 公司 PaaS（平台即服务）解决方案为快速应用开发与部署提供了支持，尤其在企业级服务、混合云策略等领域表现突出。阿里云在中国和亚太地区都具有显著的影响力，为当地的企业提供了经过优化的性能与合规性的解决策略。

2.4 集成开发环境和代码编辑器（如 Visual Studio Code, IntelliJ IDEA）

集成开发环境和代码编辑器是开发人员在日常工作中不可或缺的工具，它们直接影响到开发的效率和用户的舒适度。VS Code 是一种轻便但功能丰富的代码编辑工具，能够兼容几乎所有的主流编程语言。其插件具有丰富的生态与活跃的社区，可通过扩展对代码自动完成，版本控制，容器管理等多种功能进行辅助。VS Code 在保持较高可定制性的前提下，其简洁的接口使其成为众多开发者首选的工具。另一方面，IntelliJ IDEA 作为一个功能完善的集成开发环境，尤其适合 Java 开发环境。为开发者提供深度代码分析，智能代码补全以及系列自动化重构工具等服务，极大地促进开发者生产力。IntelliJ IDEA 具有优秀的用户体验，智能理解项目上下文。

3.互联网软件开发工具和平台比较与选择

3.1 各工具和平台的功能对比

互联网软件开发工具与平台功能宽泛而多样，为版本控制，自动化构建，云服务以及开发环境提供了大量选择。就版本控制而言，Git 与 SVN 代表着分布式与集中式两种方法论，Git 中支持离线提交与强大分支管理，但 SVN 具有直观权限控制，简单历史跟踪等功能。当谈到自动化的构建和部署时，Jenkins 和 Travis CI 都提供了自动化流水线的构建服务。然而，Jenkins 因其高度的可配置性和强大的社区插件库而受到赞誉，

而 Travis CI 则因其与 GitHub 的深度集成和易于掌握的特点而受到青睐。AWS、Azure 和阿里云都是云服务发展所长。AWS 提供了一系列的全球服务，Azure 对其与微软产品的整合进行了优化，而阿里云则在中国和亚太地区提供了高质量的服务。在开发环境方面，Visual Studio Code 和 IntelliJ IDEA 都具备代码自动完成和集成调试的能力，但 VS Code 在轻量和速度上表现得更为出色。

3.2 性能与稳定性评估

在软件开发工具与平台中，性能与稳定性是一个很重要的指标。Git 性能优越，尤其对大规模项目的处理速度与效率具有显著优势。SVN 对版本控制稳定性有很长的历史记载，特别是对于线性开发流程。Jenkins 的复杂性使得它在大型的工程中有可能遭遇性能瓶颈的问题，但是它的稳定性与成熟度却被人们普遍接受。相对而言，Travis CI 所提供的服务更轻量级，尽管表现得更优秀，但是在超大规模项目上可能会受到挑战。在云服务中，AWS，Azure 以及阿里云均进行了优化，提供了高性能以及高可靠性服务，其差别主要体现在服务可用区以及定制化选项上。

3.3 成本效益分析

Git、SVN 等开源工具的直接费用比较少，但是可能要在培训、维护等方面进行投入。Jenkins 作为一个免费、开源 CI/CD 工具其初始投入代价很高，特别是在配置、自定义插件方面，但是从长远看通过自动化节省下来的费用能够抵消掉这些代价。Travis CI 一般都是按照使用量计费，对小型项目而言也许费用更少，但是费用会随规模而提高。AWS，Azure，阿里云等云服务平台均提供了按需付费模式，可按实际用量进行成本调整，但是综合的业务及数据传输费用有可能积累为昂贵的成本。

3.4 用户体验与社区支持

用户体验与社区支持是软件开发工具与平台选择的关键。Git 拥有一个广大且充满活力的开发者社群，而 SVN 由于其直观的界面设计，特别适合那些偏好集中控制版本的团队使用。Jenkins 拥有巨大的用户基础以及插件社区并提供丰富的定制化选项以及支持资源，但是这样也会让新用户上手的流程复杂化。Travis CI 的用户界面设计得既简洁又直观，尤其是对于 GitHub 的用户来说，同时其相关文档和社区指南也是比较易于理解的。云服务平台中，AWS、Azure 等平台均拥有大量文档、论坛等支撑，阿里云为中国市场提供本地化支撑。关于开发环境，Visual Studio Code 流畅度高、可定制性强等特点为用户带来高质量体验，它强大的社区支持、丰富的插件库也是它深受欢迎的一个原因。IntelliJ IDEA 尽管要支付一定的费用，但是其卓越的智能编码特性以及卓越的设计使其在专业开发环境下拥有忠实的用户群。

4 结束语

伴随着科技的进步，敏捷开发与协同工作等技术支持工具与平台不断进化。开发团队通过对各类工具与平台在功能，性能，成本效益以及用户体验等方面进行深入评价，能够较好地筛选出符合其需求的系统。一个合适的工具组合能够有效地促进敏捷方法论的实施，提升团队协作效率，加快产品迭代周期，最终为企业带来更大的竞争优势。用户体验与社区支持在工具选择过程中同样是必须考虑的问题，因为这些问题直接关系着工具学习曲线以及问题解决效率。一句话，科学、合理的工具与平台选择是软件开发效率与质量提升的关键。

参考文献

[1] GB/T 43435-2023, 信息安全技术 移动互联网应用程序(App)软件开发工具包(SDK)安全要求[S].

[2] 贾晓辉, 李勇军. “互联网+”背景下软件开发技能学习服务支架研究[J]. 信息与电脑(理论版), 2022, 34 (03): 249-252.

[3] 刘博豪. 基于互联网的软件开发技术推广服务管理平台 V1.0. 湖北省, 武汉东湖学院, 2021-11-01.

[4] 杜彬. 基于移动互联网的软件开发技术探讨[J]. 信息系统工程, 2021, (08): 65-67.

[5] 林煊, 杨田贵. 软件开发工具 Java 编程特点及其技术分析[J]. 数码世界, 2019, (05): 61.