基于大模型的端到端自动化编程工具设计与实现

——以 Web 应用开发为例

丁禹钧
青岛工程职业学院　山东青岛　266112

摘　要：针对软件开发领域碎片化、技术门槛高的问题，提出并实现了一种基于人工智能大模型的端到端自动化编程工具，重点解决 Web 应用开发中需求分析、代码生成、测试验证到部署运维的全链路自动化问题。系统采用了四层架构设计，前端展示层、大模型层、持续集成层与基础服务层。通过融合敏捷开发方法论，实现用户故事动态解析、全栈代码生成及智能部署策略。

关键词：人工智能大模型；持续集成；敏捷开发；Web 应用开发

引言

自从 ChatGPT 发布以来，在国内外人工智能尤其是人工智能大模型发展迅速。在软件开发领域，人工智能大模型已经应用在代码生成和优化、智能化测试、代码审查、安全加固、团队协作策略优化等多个场景，但现阶段只局限于单一场景，技术门槛依然较高，未来的趋势是实现从需求分析到部署运维的端到端自动化编程，软件开发领域的分支繁杂，本文针对 Web 应用开发基于人工智能大模型尝试设计并实现了端到端的自动化编程。

1 相关技术介绍

1.1 Vue.js 框架

Vue.js 框架是一种基于 MVVM 设计模式的主流前端开发框架 [1]，其组件化的开发模式极大地提高了前端开发前端开发的灵活性，用户可以非常方便的选取和应用相应的功能模块，适合非常适合自动化开发。

1.2 Spring Boot 框架

Spring Boot 是一种 Java 后台开发框架，通过自动化配置和约定优于配置原则，大幅提升开发效率，具有自动配置、热部署、快速启动、依赖管理、模块化开发等特点，极大简化了开发、部署和运维，其“约定优于配置”的理念和完善的生态系统，让开发者从繁琐的配置中解放，专注于业务价值创造，已成为构建可靠、高效企业级应用的事实标准。[2]

1.3 Jenkins

Jenkins 作为基于 Java 构建的开源持续集成 / 持续交付（CI/CD）工具，专为自动化软件开发生命周期中的构建验证、测试执行、部署编排及运维监控等关键环节而设计。该平台凭借其模块化插件架构与可视化流水线配置，为开发团队提供了高度可扩展的自动化解决方案，有效缩短交付周期并保障软件质量。[3]

1.4 其他

Web 应用开发当中还会用到数据库、中间件、容器等，本文的自动化工具需要用到这些组成的编译、运行环境。

选择 MySQL 作为数据库软件，它是一种开源、免费、跨平台的关系型数据库，数据查询主要以结构化形式为主，具有轻量、性能优越等特点，在软件工程领域中具有统治地位的关系型数据库。

选择 Redis 作为数据缓存数据库，Redis 是一个高性能、开源、基于键值对的缓存与存储工具，Redis 基于内存对数据库进行存储，保证了数据的高效读取和写入。

Docker 作为一款采用 Go 语言构建的开源容器化平台，其核心功能在于实现应用程序的封装、分发与运行管理。通过将应用及其依赖环境打包为标准化镜像，开发者能够将其便捷部署至 Linux 服务器，以容器形式执行。该技术架构的显著优势体现在轻量化运行与资源高效利用方面。沙箱化的隔离环境支持单节点服务器同时运行数百个独立容器，为分布式应用部署提供了高密度、低负载的解决方案。

Nginx 作为一款高效能的反向代理与静态资源服务器，其核心架构设计以高可靠性和优异的请求处理效能著称。[3]

ISSN: 2661-3719(Print); 2661-3727(Online)

在 Web 服务体系中，该工具被广泛用于实现负载均衡策略与资源调度管理，通过优化网络流量分配和静态文件响应机制，显著提升分布式系统的稳定性和并发处理能力。

2 系统设计

2.1 整体架构

为实现基于大模型的端到端自动化编程工具，将其拆分为 4 层，如图 1 所示。

其中，前端展示层是用于与用户交互的人机接口，用户可以通过其输入自然语言需求，并接收来自系统的代码、提示等反馈。大模型层主要收集来自前端展示层的用户输入并产生代码，可以接入 deepseek、CodeGeeX，提供需求分析、代码生成服务。持续继承层包括代码仓库 Gitlab、持续集成工具 Jenkins 以及 Java 编译环境，这些共同组成持续集成和持续交付的管道，能够提供自动检测提交的新代码、自动编译、单元测试、自动打包和自动化部署等功能。基础服务层的作用是提供部署的测试和生产环境，每一次部署都由 Docker 创建新的独立运行环境，与不同版本的运行环境隔离，运行环境中包含 Web 应用所需要的数据库 MySQL、缓存服务 Redis 和反向代理服务 Nginx 等。

图 1 分层架构图

2.2 功能设计

本系统使用敏捷式开发方法，通过频繁的小规模交付与持续反馈，不断改进课程质量和开发流程 [4]，非常适合基于人工智能大模型的自然语言零代码开发。

如图 2 所示，将系统划分为五个功能模块，分别是用户故事分析模块、代码自动生成模块、代码自动测试模块、自动部署发布模块已经运行环境管理模块。

敏捷开发采用用户故事来阐释需求，用户故事分析模块就是将用户简短、碎片化且模糊的需求编程符合敏捷开发方法的用户故事，就是传统软件工程中需求分析。代码自动生成模块的利用人工智能大模型将用户故事转换为代码。代码自动测试模块包括单元测试和集成测试，负责将提交后的代码在相应环境中运行，测试相关接口及其交互是否正确。自动部署发布模块能够将测试后的代码自动编译、打包和部署。运行环境管理模块能够管理编译、打包和运行 Web 应用程序的环境，提供运行所需的资源。

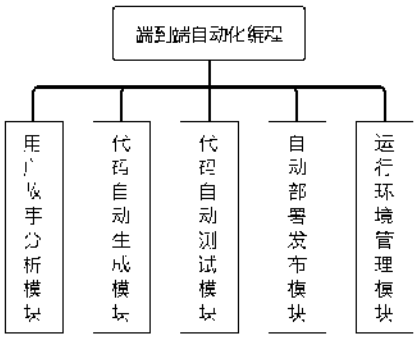

图 2 功能模块图

2.3 界面设计

系统页面主要包括用户交互页面、代码输出页面、项目管理相关页面以及系统管理相关页面。其中用户交互页面负责用户的需求输入和系统的输出，项目管理页面负责管理用户创建的 Web 应用项目，对其进行全生命周期管理，系统管理页面主要是对本系统的基本设置、运行环境、资源使用状态进行管理。

3 系统实现

3.1 大模型层实现

采用适配器模式接入多种大模型，通过统一接口屏蔽底层差异。设计分层提示模板，将用户需求转换为结构化提示链，包括需求解析层、架构设计层、代码生成层，提取用户故事中的实体、关系和约束条件，生成技术选型建议，分模块生成前端组件、后端服务和测试用例。

3.2 持续集成 / 交付流水线

基于 Jenkins Pipeline 实现全自动化 CI/CD 流程，部分代码如下：

```
pipeline {
    agent any
    stages {
        stage(‘checkout’) { steps { checkout scm } }
```

```
            stage(‘anal’) { steps { sh ‘npm run lint && mvn sonar:sonar’ } }
            stage(‘unit-test’) { steps { sh ‘npm test && mvn test’ } }
            stage(‘build-package’) { steps { sh ‘npm build && mvn package’ } }
            stage(‘’) { steps { sh ‘docker-compose up -d && mvn verify’ } }
            stage(‘deploy-dev’) { when { branch ‘develop’ }; steps { sh ‘deploy-to-staging.sh’ } }
            stage(‘deploy-pro’) {
              when { branch ‘master’ };
              steps {
                input message:‘确认部署生产环境？’, ok:‘确认’
                sh ‘deploy-to-production.sh’
              }
            }
        }
        post {
          success { notify(‘SUCCESS’) }
          failure { notify(‘FAILURE’) }
    }}
```

3.3 前端展示层实现

前端展示层基于 Vue.js+Electron 构建跨平台界面，使用 Element UI 库和 Monaco Editor 等 UI 组件，完成包括支持 Markdown 编辑的自然语言交互界面，可编辑的可视化代码编辑界面，通过 Jenkins API 获取构建日志，来展示项目全生命周期管理。

3.4 运行环境管理

运行环境管理模块采用三层弹性架构实现环境自动化管理。编排层基于 Kubernetes 与 Docker Swarm 实现容器集群调度，支持跨主机资源池化管理。服务层集成 MySQL、Redis、Nginx 等中间件的定制化镜像，提供版本化环境模板。控制层通过 RESTful API 接收上层指令，结合 Prometheus 指标动态调整资源配置。使用 Kubernetes 对 docker 容器进行编排，基于 Ansible+Terraform 进行配置管理，使用 Prometheus + Grafana + Loki 对运行环境进行监控。

4 总结

本研究设计并实现了基于大模型的端到端自动化编程工具，实现从自然语言需求到生产部署的全链路自动化，降低重复开发工作。通过统一接口支持多种大模型，可根据场景动态选择最优模型。将大模型技术与敏捷开发方法论深度结合，支持需求快速迭代，各层之间松耦合设计，支持后续功能模块的灵活扩展。

尽管系统取得了一定成果，但仍存在一些局限。首先复杂业务场景支持不足，对涉及复杂业务规则和算法的场景，生成代码的准确率有待提高。目前仅支持基础部署，对复杂运维场景支持有限。如果能够解决这些问题，本工具将显著降低软件开发门槛，可有效缓解人力压力，提升软件产业整体效率。

参考文献：

[1] 王思辰 , 李林 . 基于 Vue.js 的电商管理平台的设计与实现 [J]. 现代信息科技 ,2021,5(14):13-15+20.

[2] 孔倩 . 基于 Spring Boot+Vue 框架的阅览室现刊查询系统设计与实现——以上海图书馆为例 [J]. 现代信息科技 ,2025,9(07):98-102+108.

[3] 应毅 , 刘亚军 , 俞琰 . 利用 Docker 容器技术构建大数据实验室 [J]. 实验室研究与探索 ,2018,37(2):264-268.

[4] 徐慧芬 , 庞畅 , 郑如歆 , 等 . 生成式人工智能赋能敏捷课程开发与实践研究 [J]. 远程教育杂志 ,2024,42(05):95-101.

作者简介： 丁禹钧 (1993 —) , 男 , 汉族 , 山东省青岛市 , 助教 , 硕士 , 研究方向：计算机应用技术。