文章编号:1671-4598(2025)06-0193-07 DOI:10.16526/j.cnki.11-4762/tp.2025.06.024 中图分类号:TP311 文献标识码:A

试飞流程引擎平台软件设计与应用

陈家益，刘 涛，冯 灿，李成浩，薛博文
（中国商飞民用飞机试飞中心，上海 200232）

摘要：为了进一步推进试飞业务数字化转型进程，建立符合试飞特色的数字化应用平台，设计研发了一套试飞流程引擎，采用了前后端分离的架构，集成 Activiti 流程引擎实现试飞管理业务流程的在线编辑，修改和管理，提出“代码自动生成”的低代码构想，引入 Form Generator 组件，根据用户选择的模板和参数，自动生成实体类、控制器、服务类以及前端页面代码，同时，系统还设计了健全的消息通知，权限管理，日志记录等功能，方便开发人员更加聚焦业务功能实现；经过测试，该系统实现了超过 50 人的同时访问和使用，低代码功能将一天的开发量缩短至半个小时，该平台具备良好的可扩展和可移植性，为试飞数字化转型提供了有力的技术支持。

关键词：数字化转型；试飞；Activiti；低代码；系统

Design and Application of Flight Testing Process Engine Platform Software

CHEN Jiayi, LIU Tao, FENG Can, LI Chenghao, XUE Bowen
(Flight Test Center of COMAC, Shanghai 200232, China)

Abstract: To further advance the digital transformation process of flight testing operations and establish a digital application platform that aligns with the unique characteristics of flight testing, a flight testing process engine was developed. This engine utilizes a front-end and back-end separation architecture and integrates the Activiti process engine to achieve the online editing, modification, and management of flight testing business processes, presents a low-code approach with code auto-generation, introduces a Form Generator component, and automatically creates entity classes, controllers, service classes and front-end page code based on user-selected templates and parameters. Additionally, the system has the functions of message notification, permission management, and log recording, it is convenient for developers to implement business functions. The system supports simultaneous access and usage by over 50 users, significantly reducing its low-code feature and shortening the development time from a day to half an hour. The platform also has excellent scalability and portability, providing a strong technical support for the digital transformation of flight testing.

Keywords: digital transformation; flight testing; Activiti; low-code; system

0 引言

数字化是当代社会发展的重要趋势，它通过运用先进的信息技术，将传统业务转化为数字形式，实现数据的高效处理、传输和应用。数字化不仅提升了工作效率，还推动了行业创新，成为推动社会进步的关键力量[1-3]。

随着航空科技的日新月异，新型飞机设计与制造技术的不断进步，试飞业务面临着越来越高的要求。传统的试飞方式往往依赖于人工操作和纸质记录，这种方式不仅效率低下，而且容易出错，难以满足现代航空工业的发展需求。同时，随着信息技术的快速发展，数字化转型已经成为航空工业发展的必然趋势，试飞业务数字化也成为行业发展的重要方向[4-5]。

随着低代码和流程引擎技术的发展，数字化转型也有了新的发展方向，例如国内成熟 RuoYi 等低代码开发框架强大的代码生成器，让前后端代码一键生成，减少重复工作，提高开发效率，降低门槛。

随着国内某大型民机成功投入商业运营，国产民机已正式迈入规模化和系列化发展的新纪元。然而，伴随而来的国产民机安全管理形势亦日益严峻，不容忽视。当前，试飞安全业务管理过程中凸显出若干亟待解决的痛点[6]：

1）试飞组织管理体系由一本顶层手册、若干程序

收稿日期:2024-04-22； **修回日期**:2024-05-15。

作者简介:陈家益(1990-),男,硕士,工程师。
通讯作者:冯 灿(1983-),男,博士,研究员级高级工程师。

引用格式:陈家益,刘 涛,冯 灿,等.试飞流程引擎平台软件设计与应用[J].计算机测量与控制,2025,33(6):193-199.

文件以及若干过程表单组成。然而，当前的管理手段过于落后，主要依赖于纸质文档进行流转，这种方式不仅管理过程复杂，而且效率低下，难以适应现代试飞组织管理的高效需求。

2）当前的试飞工作呈现出多基地、多型号、多架机的复杂趋势。在这种背景下，现有的试飞业务管理模式显得力不从心，难以满足试飞安全快速响应和高效同步的迫切需求。

因此，采用先进的信息化技术，深度结合当前试飞组织管理工作的实际状况开展试飞流程引擎平台产品设计与开发工作。

1 系统结构及原理

试飞流程引擎平台集成低代码功能和流程引擎功能，通过自定义表单和流程接口，支持表单和流程的自动关联和流程发起，实现一套具备试飞特色的流程引擎平台。

试飞流程引擎平台采用 B/S 架构，实现了前后端彻底分离技术，同时结合高并发数据库构建试飞安全管理系统，实现相关流程、表单的电子化，保证基础安全信息的规范完整，实现各类安全事件的上报和审批流程的可视化，提升安全管理信息化效能，推动试飞管理由纸质化、人工化走向信息化、数字化。软件结构如图 1 所示，软件的 B/S 结构如图 2 所示。

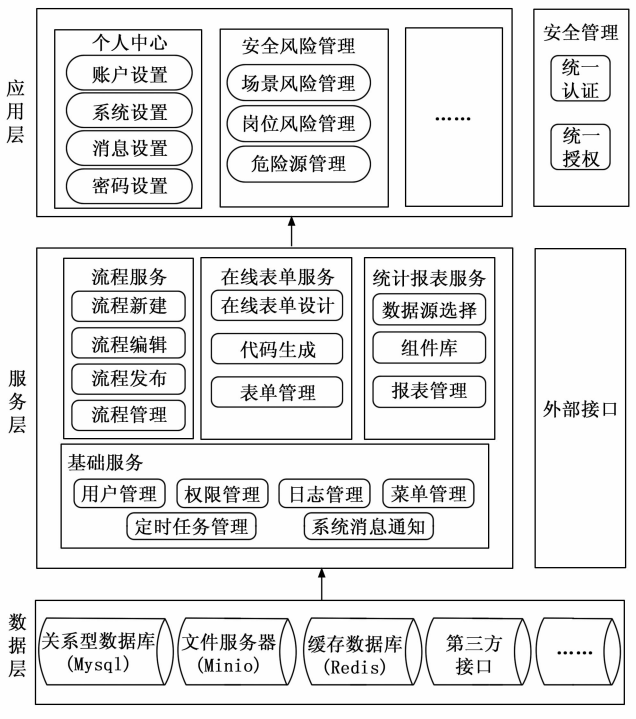

图 1 系统软件结构示意图

本系统选择采用主流的 Vue＋SpringBoot 框架[7]进行开发，提出数据层、服务层和应用层共 3 层架构：

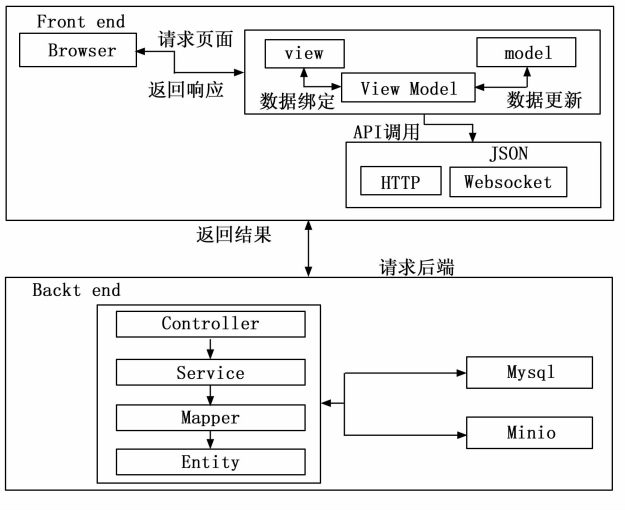

图 2 软件的 B/S 结构

1）数据层包含关系型数据库、非关系型数据库、缓存数据、文件服务器及外部接口等。

2）服务层包含基础服务、流程服务、表单服务、在线表单服务和统计报表服务；

3）应用层包括个人中心、权限、角色、通知、日志以及各类可扩展的业务功能模块。

用户通过公司内网浏览器即可登录系统。本系统前后端之间的通信主要通过 HTTP（HTTP，hyper text transfer protocol）和 WebSocket 两种方式实现。HTTP 以请求－响应模型，高效处理表单、图片、文件等一次性请求；而 WebSocket 则以其全双工通信能力，实现客户端与服务器间的实时、双向数据传输，尤其适用于消息推送等实时性要求较高的场景。

2 后端架构设计

后端服务以 SpringBoot 框架为核心，集成 Mybatis-plus 组件，实现对数据库的灵活增删改查操作，从而支持结构化和半结构化数据的持久化管理。为确保系统安全性与灵活性，采用 shiro 框架，实现用户、菜单、权限的精细分级管理。此外，还引入了日志框架，对系统运行状态和用户操作进行实时监控与存储，确保系统的稳定运行与操作的透明性。

2.1 数据访问设计

本系统采用 MyBatis-plus 作为数据访问工具，MyBatis-plus 支持常用的数据库查询，是存储过程和高级映射的主流持久层框架[8]。MyBatis-plus 通过使用 XML 或注解进行简单配置，便可快速进行单表增删改查操作，并且支持代码生成、物理分页、性能分析等功能，整个集成和数据访问工作流程如图 3 所示。

2.2 数据存储设计

本系统涉及结构化数据和非结构化文本数据，其

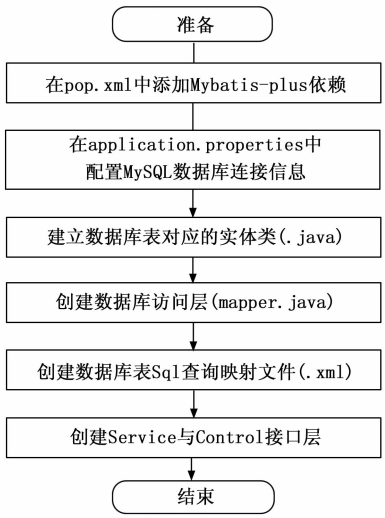

图 3　集成和数据访问流程图

中：结构化数据全部存储在关系型数据库 MYSQL 中，文本数据存储在本地搭建的 MINIO 文件服务器[9]中。

1）结构化数据：本系统的数据库命名以字母“db”开头（小写），后面加数据库相关英文单词或缩写，如表 1 所示。

表 1　数据库命名

数据库名称	描述
db_sms	安全管理系统数据库

本系统共分四类表：第一类为流程引擎表，第二类为在线表单引擎表，共 10 张表，第三类为系统基本功能表，包括人员信息、权限和日志等共 15 张表，主要表见表 2，第四类为业务表，通过在线表单生成或自定义新建。

表 2　系统基本功能主要表

数据表名称	描述
sys_user	用户基本信息表
sys_depart	部门信息
sys_role	角色表
sys_user_role	用户角色表
sys_role_permission	用户角色权限表
sys_permission	用户权限表
sys_log	系统日志
sys_dic	系统字典
sys_announcenent	系统消息通知

2）文本数据：本系统涉及大量的图片，文本等过程文件上传和保存，考虑到文件的安全和管理方便，选择内网搭建 MINIO 文件服务器，MINIO 是一个高性能、分布式的开源对象存储系统，适用于存储和分发多媒体内容，如图片、视频和音频文件。MINIO 数据存储路径见图 4，整个文档存储目录分三层：第一层为系统层，第二层为模块层，第三层为文件层。为保证文件的唯一性，在上传文件时，系统会自动在文件名添加一个时间戳。

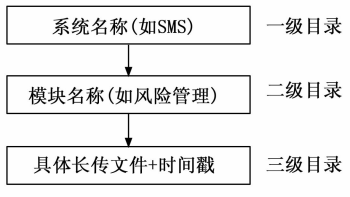

图 4　MINIO 数据存储目录

2.3　系统接口设计

系统接口设计包括内部接口设计和外部接口设计。

1）外部接口：本系统对外主要对接 OA 系统和试飞业务系统接口。其中：试飞业务主要用于接入现有的试飞相关业务数据；OA 系统主要用于用户的单点登录，将 OA 系统的用户信息及组织架构信息同步到本系统。

2）内部接口：本系统内部接口主要有两种，一种是采用 HTTP 协议的 RESTful 接口，还有一种是 Web Service 的消息推送接口。两种接口的数据通讯格式都是采用标准的 JSON 格式。

本系统采用 Swagger 进行接口管理，Swagger 是一个规范和完整的框架，用于生成、描述、调用和可视化 RESTful 风格的 Web 服务。Swagger 可以自动生成 API 文档，通过 Swagger UI 中，可以浏览、测试和查看 API 的详细描述、请求和响应的示例等。

2.4　权限管理模块

权限管理采用经典的 RBAC（Role Based Access Control）方式进行控制，RBAC 的核心就是通过角色来控制用户权限，将权限赋值给角色，角色赋值给用户，来实现对系统资源的访问控制[10]。

如图 5 所示，权限管理涉及用户、菜单及角色等，系统将建立用户管理、部门管理、岗位管理、菜单管理及角色管理等模块功能。

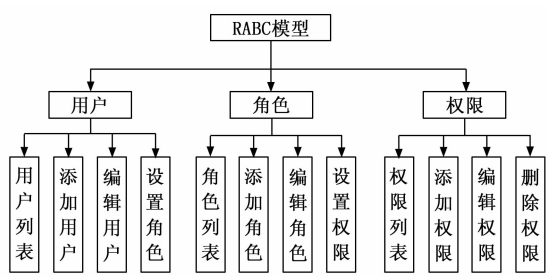

图 5　用户功能权限管理流程图

2.5　日志信息模块

日志是系统运行轨迹的忠实记录，详尽记载了软件在运行过程中的关键行为、状态变化和异常事件，对于排查问题、优化性能、审计追踪、安全防护等方面具有

不可替代的价值，是保障系统稳定、安全、高效运行的核心支撑之一。

本软件使用 Logback 和 Log4j 作为日志实现框架，将日志输出到控制台和文件系统，同时可以根据配置将日志发送到数据库进行存储，便于日志的持久化存储和检索。日志系统支持多种日志级别，如 TRACE、DEBUG、INFO、WARN、ERROR、FATAL 等，可以根据实际需求调整日志输出的详细程度。

3 前端架构设计

本系统前端框架选用 Vue[11-12]，Vue 专注于视图层的构建，凭借渐进式的设计理念和出色的性能，无论是单页应用、移动应用还是复杂的 Web 应用程序都表现出色。Vue 支持 Vue Router、Vuex、Vue Devtools、Axios 等工具，使得能够构建出高效、可维护的 Web 应用程序。Vue 的 MVVM 模型将数据模型（Model）与视图（View）解耦，通过视图模型（ViewModel）协调通信，数据驱动视图，避免直接操作 DOM，使前端开发更高效便捷。

3.1 接口管理模块

本系统采用 Axios 进行前端接口管理，Axios 作为一款基于 Promise 的 HTTP 库，支持同步或异步的 HTTP 请求。在交互过程中，前端通过 Axios 发送 HTTP 请求至后台 controller 控制器，controller 接收请求后，会调用相应的 service 层处理业务逻辑。若涉及数据库操作，service 层会进一步调用 mapper 执行 SQL 语句。后台处理完毕后，以 JSON 格式返回数据给前台，前台则解析 JSON 并展示给用户。

Axios 功能强大，提供了灵活的请求和响应转换、错误处理及认证信息添加机制。其中，拦截器功能尤为出色，包括请求拦截器和响应拦截器两部分。请求拦截器在请求发送前发挥作用，可用于添加 token、设置请求头或通用参数等；而响应拦截器则对接收到的响应数据进行处理，如转换格式、错误处理等。这些特性使得 Axios 成为前端接口管理的理想选择，有效提升了系统的稳定性和用户体验。

3.2 菜单管理模块

本系统采用 Vue-router 进行路由和菜单管理，通过 Vue Router 建立前端页面的路由系统，实现页面间的跳转和切换。Vue Router 可以将不同的 URL 映射到相应的 Vue 组件，并在页面切换时实现组件的动态加载和渲染。

本系统路由分为静态路由和动态路由，静态路由是指不需要进行权限控制的页面，动态路由则需要进行权限动态加载。系统可以通过前端页面进行菜单的配置，并通过角色将菜单权限赋值给对应的角色。

3.3 报表管理模块

安全管理系统作为重要的质量管理数字化系统，将重要的安全生产指标以报表的形式进行综合展示显得非常重要。本系统采用外部 TempoBI 报表工具进行系统关键综合指标展示需求的开发工具。TempoBI 支持 50 余种图形的可视化展示，包括近 20 种高级图形组件、10 余种辅助组件、3D 图形组件等，可以满足用户多样化的展示需求。TempoBI 开发流程如图 6 所示。

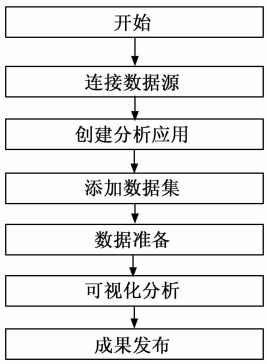

图 6 基于 TempoBI 开发流程图

3.4 通知模块设计

消息通知模块主要是进行相关通告，流程等信息的编辑，推送和管理。通知管理包括模板管理，消息管理和我的消息中心。模板管理：可以新建消息发送模板，模板类型包括邮件，系统等类型，可以自定义编辑模板内容。消息管理：可以查看消息方式状态。我的消息中心是一个集中式通知平台，它汇集了用户的各类互动信息，如流程动态、系统通知等，旨在让用户一站式查阅和管理全部消息，实现高效沟通与信息同步，提升用户体验与社交效率[13]。

4 流程引擎模块设计

试飞流程引擎平台涉及大量的业务流程和过程表单，为了保证业务流程自动化、可视化以及版本管理，同时支持跨系统集成，保障业务连续性和稳定性，本项目采取引入 Activity 流程引擎工具[14-16]实现对流程的在线编辑，管理和版本管理，以应对复杂多变的流程需求。

本系统的流程引擎会与系统的低代码平台会形成一套具备试飞特色的表单和流程引擎，通过表单配置实现表单和流程的关联和自动启动。

4.1 Activity 核心组件

Activity 流程引擎重点在系统开发的应用性和轻量性上，流程引擎以服务的形式提供给服务调用人员。服务调用人员可以构建丰富、轻便且高效的应用程序，Activity 系统服务结构如图 7 所示。

Activiti Engine 是一款功能强大的业务流程管理引擎，它全面支持 BPMN 2.0 规范，能够实现流程的解

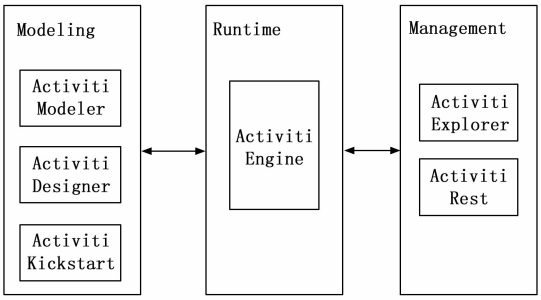

图 7　Activity 流程引擎结构图

析、执行、创建与管理，包括任务分配、流程实例监控等。Activiti Modeler 则作为模型设计器，帮助业务人员将业务需求转化为标准化的流程定义。Activiti Designer 则提供了可视化的设计功能，支持将 XML 格式的流程定义导入并加工成可运行的流程。Activiti Explorer 则用于管理仓库、用户、组等，并支持流程的启动与任务办理。而 Activiti REST 则提供了 RESTful 风格的服务，允许客户端以 JSON 方式与引擎交互，实现跨平台、跨语言的灵活应用。

4.2　Activity 服务

Activiti 引擎作为一款强大的业务流程管理（BPM）开源框架，提供了多个服务来支持流程定义、部署、执行、监控以及管理。以下是 Activiti 引擎主要的服务组件：

1）仓库服务（RepositoryService）：确保业务模型的有序存储与检索。通过该服务，用户可以轻松上传、部署、删除流程模型，并实时查看版本信息，从而实现对业务流程模型的精细管理与版本控制。

2）运行服务（RuntimeService）：主要管理和控制正在运行的流程实例。该服务可以启动新的流程实例、查询现有实例、挂起和激活实例等功能。

3）任务服务（TaskService）：主要负责任务级别的操作，包括创建任务、查询任务列表、完成任务、指派任务、更新任务等。

4）表单服务（FormService）：考虑到 Activiti 表单功能比较单一，故本系统引入一套新的表单生成工具，并重构表单和流程接口，实现业务表单和流程引擎的关联。

5）历史服务（HistoryService）：主要提供了查询流程实例的历史数据、历史任务、审计信息等功能，便于对历史流程执行情况进行追溯和分析。

6）身份服务（IdentityService）：本系统开发通用权限和认证功能模块，不使用该身份服务。

7）管理服务（ManagementService）：提供了更多的后台管理和维护功能，如清理历史数据、获取数据库表信息、执行 SQL 查询、获取作业执行信息等。

8）流程引擎（ProcessEngine）：作为 Activiti 引擎的核心，它集成了上述所有服务，是其他所有服务的基础，提供了对业务流程生命周期的整体控制。

4.3　数据库表结构

Activiti 采用了分表设计，将流程相关的数据分布在不同的数据库表中，以便有效地管理和存储流程定义、运行时数据、历史数据以及其他辅助信息。Activiti 流程引擎数据表分别如表 3～7 所示。

表 3　一般数据表（ACT_GE_）

ACT_GE_BYTEARRAY	二进制数据表，存储通用的流程定义和流程资源。
ACT_GE_PROPERTY	系统相关属性，属性数据表存储整个流程引擎级别的数据，初始化表结构时，会默认插入三条记录。

表 4　流程历史记录（ACT_HI_）

表名	解释
ACT_HI_ACTINST	历史节点表
ACT_HI_ATTACHMENT	历史附件表
ACT_HI_COMMENT	历史意见表
ACT_HI_DETAIL	历史详情表，提供历史变量的查询
ACT_HI_IDENTITYLINK	历史流程人员表
ACT_HI_PROCINST	历史流程实例表
ACT_HI_TASKINST	历史任务实例表
ACT_HI_VARINST	历史变量表

表 5　用户用户组表（ACT_ID_）

表名	解释
ACT_ID_GROUP	用户组信息表
ACT_ID_INFO	用户扩展信息表
ACT_ID_MEMBERSHIP	用户与用户组对应信息表
ACT_ID_USER	用户信息表

表 6　流程定义表（ACT_RE_）

表名	解释
ACT_RE_DEPLOYMENT	部署信息表
ACT_RE_MODEL	流程设计模型部署表
ACT_RE_PROCDEF	流程定义数据表

表 7　运行实例表（ACT_RU_）

表名	解释
ACT_RU_EVENT_SUBSCR	运行时事件 throwEvent、catchEvent 时间监听信息表
ACT_RU_EXECUTION	运行时流程执行实例
ACT_RU_IDENTITYLINK	运行时流程人员表，主要存储任务节点与参与者的相关信息
ACT_RU_JOB	运行时定时任务数据表
ACT_RU_TASK	运行时任务节点表
ACT_RU_VARIABLE	运行时流程变量数据表

Activiti 通过这些表结构实现了流程定义的存储、

流程实例的运行管理、用户与权限的控制以及流程执行的历史记录，使得业务流程在系统中的全生命周期得以高效、有序地管理和维护。在流程实例结束后，Activiti 会将不再需要的运行时数据从 ACT _ RU _ 表中删除，以保证这些表的小巧和高效。

4.4 流程设计器

本系统集成 BPM（Business Process Management，业务流程管理）作为流程设计器，主要负责解释、执行和管理业务流程模型[17]。如图 8 所示，流程引擎依据预定义的流程图或流程模型来驱动业务流程的执行，主要包括以下几个阶段：

1）流程图定义阶段：首先，使用 BPMN（Business Process Model and Notation，业务流程模型和标记法）通过流程图描述流程的各个步骤（活动）、它们之间的连接关系（序列流、并行网关、Exclusive Gateway 等）、以及流程中可能出现的事件和条件。

2）流程部署阶段：将设计好的流程图导入或部署到 BPM 流程引擎中，流程引擎会对流程模型进行解析和验证，确保其符合规范并可执行。

3）流程执行阶段：

（1）流程实例化，当需要启动一个业务流程时，流程引擎会根据流程图创建一个新的流程实例。

（2）活动调度，流程引擎根据流程图的定义，按顺序或条件触发流程实例中的各个活动。例如，当一个任务完成后，流程引擎会根据流程流转逻辑判断下一个要执行的任务或分支。

（3）任务分配与执行，流程引擎会根据预先定义的角色、用户组或动态规则将任务分配给相应参与者。参与者完成任务后，流程引擎会接收任务完成的信号，并继续推进流程。

4）结束阶段：当流程达到正常或异常结束条件时，流程引擎会完成所有必要的清理操作，关闭流程实例，并生成最终的结果数据或报告。

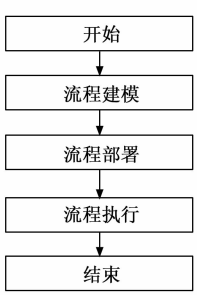

图 8 流程引擎工作流程

5 低代码模块设计

低代码平台[18-20]是推动数字化转型、赋能快速创新和敏捷开发的重要工具。考虑本系统有大量功能相似的管理页面需要开发，任务重、工作量大，因此，本系统引入低代码模块，旨在通过显著减少手工编码的工作量来加快应用程序的开发速度和简化软件构建过程。

本系统的低代码平台会形成一套表单系统，通过与流程引擎接口进行对接，形成一套具备试飞特色的表单和流程引擎。

5.1 前端页面设计

本系统集成 VUE 开源 Form-Generator 表单生成工具，本工具包括组件库、页面设计及代码生成 3 个部分。

1）组件库：组件库包括输入型组件，选择型组件和布局型组件等 19 个常用功能组件，覆盖基本页面使用需求，并且还支持进行组件的自定义扩展。

2）页面自动设计：通过本工具可以快速拖拉拽方便快捷的搭建所需页面，针对具体组件可以进行组件属性的自定义，包括字段名，标题，表单栅格等共 15 项属性设置，并且可以根据需要设置组件正则校验信息。

3）前端代码自动生成：页面生成类型包括页面或弹窗类型，开发人员可通过选择页面生成类型定制化生成所需代码。

5.2 后端代码设计

后端代码设计基于 ruoyi-generator 实现，包括数据表结构设计和代码生成。

1）后端表结构设计：支持在线建表，表单修改，以及支持一对一、一对多及多对多等关系型数据库的设计和建表功能。

2）代码生成模块：新建的表支持查看、修改、删除、同步和生产代码等功能，可以在线生成所需后端代码和数据库代码。

6 试验结果与分析

本系统从系统性能和系统功能两方面进行功能测试。

6.1 系统性能试验

系统性能试验主要通过将该系统成功部署到公司内网开始进行，并经过一个月的试运行，累计注册用户、表单统计、流程统计、同时在线人数及宕机数，见表 8，该平台总体运行稳定，支持不少于 50 人同时在线操作，同时在线表单和在线流程具备高可扩展性，能满足大部分业务场景需求。

表 8 系统运行情况统计

累计注册用户/人	表单统计/人	流程统计/人	同时在线人数/人	宕机次数
1 218	56	28	54	0

6.2 系统功能试验

系统功能试验主要包括低代码功能、流程引擎功能

以及消息通知功能。本次试验以具体业务需求为牵引，将业务需求转化为功能需求，并通过平台进行具体实现，主要包含以下 4 步：

1）完成业务需求结构化，进行数据库设计，并通过平台建立数据库表结构，包含一对多等复杂表类型，其清单如表 9 所示。

表 9　数据库表结构

表名	备注
risk_post_dangerous	场景风险源
risk_scene_dangerous	岗位风险源
risk_consequence	后果库
risk_dangerous_control	风险措施表

2）代码功能：通过平台快速生成前后端代码和管理页面。

3）根据业务需求，设计业务流程审批模型。

4）通过消息通知进行及时通知和审批，实现对业务模块的管理和应用。

经过该业务模块的开发和测试，本系统能较快地支持多表的新建，代码自动生成，通过流程引擎实现流程的编辑，管理和执行，实现该大部分业务功能。

7　结束语

本文设计研发了一套高效的流程引擎平台，支持前后端代码自动快速生成，将代码开发效率由一天缩短到半个小时，实现 50 人的同时访问和使用，该平台具备良好的可扩展和可移植性，为试飞数字化转型提供了有力的技术支持。

参考文献：

[1] 李永涛. 中小企业数字化转型的思路与步骤 [J]. 江苏商论，2024 (4)：24－26.

[2] 陈德球，张雯宇. 企业数字化转型与产品市场竞争地位 [J]. 武汉大学学报（哲学社会科学版），2024，77 (2)：118－131.

[3] 佟　岩，刘柯慧，赵泽与，等. 企业数字化转型对客户集中度的影响 [J]. 北京理工大学学报（社会科学版），2024，26 (1)：177－194.

[4] 谢　康，肖静华，王刊良，等. 企业高质量数字化转型管理：理论前沿 [J]. 管理学报，2024，21 (1)：1－9.

[5] 王长兴，张敬宇，牛勇力. 企业数字化转型中的低代码开发平台应用 [J]. 电子技术，2023，52 (11)：334－335.

[6] 戴宏毅. 试飞安全管理体系与质量管理体系的融合研究 [J]. 机械制造，2020，58 (1)：86－89.

[7] 徐少军，李宗哲，梅　杰，等. 基于 Springboot＋Vue 框架的质量检验监督管理系统研发 [J]. 纺织标准与质量，2024 (1)：11－14.

[8] 叶　刚，王立河，王英明，等. 基于 Mybatis Plus 的动态生成代码设计与实现 [J]. 电脑编程技巧与维护，2019 (7)：7－8.

[9] 李　兵，王连忠，司运成. MinIO 存储在监拍系统中的应用设计 [J]. 工业控制计算机，2020，33 (9)：79－80.

[10] 刘建华，任丹丹. 一种基于访问控制的敏感信息分类授权模型 [J]. 西安邮电大学学报，2023，28 (3)：63－67.

[11] 张　威. 基于 Vue.js 的专利学习平台开发设计与实现 [J]. 电脑知识与技术，2023，19 (22)：57－60.

[12] 江家龙. 基于 Vue.js 框架的“食在南”WebAPP 前端设计与实现 [J]. 轻工科技，2024，40 (1)：117－120.

[13] 李毓丽，陈泰宏. 基于自定义 token 消息通知系统的设计与实现 [J]. 电脑知识与技术，2020，16 (20)：84－86.

[14] 张　朋，杨鹤标. 基于 Activiti 的教学过程控制系统设计与实现 [J]. 软件导刊，2018，17 (10)：102－105.

[15] 李英吉，李　斌，陈毓春，等. 工作流引擎在电力信息化中的应用与设计 [J]. 电力信息化，2010，8 (11)：47－50.

[16] 张　剑，孟　波. 基于规则引擎的一种智能工作流系统研究 [J]. 计算机工程与设计，2006 (14)：2591－2593.

[17] 于永泽，赵　醴. 基于 BPM 企业信息系统开发平台设计分析运维 [J]. 软件，2023，44 (11)：151－153.

[18] 王长兴，张敬宇，牛勇力. 企业数字化转型中的低代码开发平台应用 [J]. 电子技术，2023，52 (11)：334－335.

[19] 许国磊. 低代码平台的价值和建设思路 [J]. 软件和集成电路，2023 (12)：19－21.

[20] 徐　辉. 基于低代码的数智化融通研究 [J]. 长江信息通信，2024，37 (1)：44－47.

（上接第 192 页）

[16] HAN K，WANG Y，TIAN Q，et al. Ghostnet：more features from cheap operations [C] //Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition，2020：1580－1589.

[17] ANDREW G，MENGLONG Z. Efficient convolutional neural networks for mobile vision applications [J]. Mobilenets，2017，10：151.

[18] ZHANG X，ZHOU X，LIN M，et al. Shufflenet：An extremely efficient convolutional neural network for mobile devices [C] //Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition，2018：6848－6856.

[19] 戴　硕，白　涛，李东亚，等. 基于知识蒸馏及改进 ShuffleNet v2 的棉花病虫害识别方法 [J]. 江苏农业科学，2024，52 (15)：222－232.

[20] BODLA N，SINGH B，CHELLAPPA R，et al. Soft-NMS-improving object detection with one line of code [C] //Proceedings of the IEEE International Conference on Computer Vision，2017：5561－5569.