计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2022,31(10):72−79 [doi: 10.15888/j.cnki.csa.008714]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

面向电力系统的可视化页面编辑引擎①

陈月燕[1,2], 邹　庆[1,2], 黄　昆[1,2,3]

[1](南瑞集团有限公司 (国网电力科学研究院有限公司), 南京 211106)
[2](国电南瑞科技股份有限公司, 南京 211106)
[3](智能电网保护和运行控制国家重点实验室, 南京 211106)
通信作者: 陈月燕, E-mail: chenyueyan@sgepri.sgcc.com.cn

摘　要: 针对电力系统前端技术多样性和手工编码方式效率低无法满足需求的快速增长的问题, 结合组件化设计思想、可视化和虚拟 DOM 技术, 设计实现了一种面向电力系统的可视化低代码页面编辑引擎. 页面构建过程中采用虚拟 DOM 技术, 适用于绝大部分场景页面的高性能渲染, 设计了统一数据模型实现异构多源数据的统一接入和多组件数据共享, 并提出基于多类型模板的页面编辑思想, 满足业务系统多样化集成需求. 实践结果表明, 该系统能够提供低代码、低门槛的敏捷高效开发, 大大地提高了电力系统前端页面的开发效率.

关键词: 可视化; 组件化; 虚拟 DOM; 统一数据模型; 模板编辑

引用格式: 陈月燕,邹庆,黄昆.面向电力系统的可视化页面编辑引擎.计算机系统应用,2022,31(10):72–79. http://www.c-s-a.org.cn/1003-3254/8714.html

Visualization Page Editing Engine for Power System

CHEN Yue-Yan[1,2], ZOU Qing[1,2], HUANG Kun[1,2,3]

[1](NARI Group Co. Ltd. (State Grid Electric Power Research Institute Co. Ltd.), Nanjing 211106, China)
[2](NARI Technology Co. Ltd., Nanjing 211106, China)
[3](State Key Laboratory of Smart Grid Protection and Control, Nanjing 211106, China)

Abstract: Regarding the power system, front-end technologies are diverse at present, whereas the manual coding is inefficient and cannot meet the rapid growth of demand. In response, this study designs a low-code visualization page editing engine for the power system with visualization and virtual DOM technologies in light of the component-based design idea. The virtual DOM technology is applied to the high-performance rendering of most scenes during page building. A unified data model is designed to integrate heterogeneous data and share data among components. The idea of a multi-type template for page editing is proposed to meet the requirement of the business system for diversified integration. The practice indicates that the system can provide agile, efficient, and low-code development under low barriers, which significantly improves the development efficiency of front-end pages.

Key words: visualization; componentization; virtual DOM; unified data model; template editing

电力行业是关系到国计民生的基础性行业, 是其他行业的基础保障, 从电力生产、输、配、变、营销以及管理等各个环节, 从传统电力系统到新型电力系统的建设转型, 电力信息化的需求越来越多. 就目前的电力系统前端领域来说, Web 端页面代码还是依靠前端开发者手工编写, 开发方式存在效率低下、技术门槛高等问题, 极大地制约了开发者的工作效率. 尤其是前端技术的飞速发展, 各种技术和框架如雨后春笋般不断地涌出, 开发难度和复杂度在逐年的增加, 这就要求开发人员即要不断学习新知识以适应技术的革新,

① 收稿时间: 2021-10-12; 修改时间: 2022-01-20; 采用时间: 2022-02-18; csa 在线出版时间: 2022-06-17

又要不断地按照用户的需求更新功能、维护已有的产品功能, 同时还要不断地应对电力系统业务需求的不断变更, 往往一个系统就需要十几人甚至上百人共同开发维护. 因此, 如何减少重复开发、缩短迭代周期、降低开发难度, 让非专业前端技术人员也能快速实现页面的开发成为了企业思考的一个重要方向. 国内已有不少这方面的探讨和研究, 文献 [1–4] 设计实现了可视化页面编辑软件开发工具, 实现了针对特定类型业务场景比如表单、Echarts 图表等的可视化编辑, 避免了其项目中反复开发一些重复性高的功能, 提高了开发效率. 而电力系统中可视化平台软件的研究主要集中在电力系统专用图形比如接线图、潮流图等方面[5–7].

目前前端业务场景主要在自动化、新能源、物联网、信息化管理、监测运维系统等方面, 相较于金融、交通等行业的业务场景, 涉及的范围更广、数据量更大、数据类别范围更广、稳定性更强, 但是电力系统之间存在很大的关联性, 数据、服务等资源可以实现共享, 并且与其他行业也存在很大的关联性. 因此, 实现资源的积累是应对电力系统需求变更, 做出快速响应的一种有效方法. 国家电网有限公司于 2021 年 4 月 15 日在北京召开第二季度工作会议, 明确提出“一体四翼”发展布局, 电网业务要立足构建以新能源为主体的新型电力系统[8], 服务碳达峰、碳中和大局. 在能源革命和数字革命相融并进, 以及“双碳”目标的大背景下, 以新能源为主体的新一代电力系统[9] 正在逐步形成. 在此背景下, 为支持新一代电力系统的快速构建和传统电力系统的信息化数据的可视化展示、管理与在线运行监测, 减少重复开发、提高组件和服务的复用性, 实现资源共享和敏捷高效的开发, 降低开发门槛. 通过对目前诸多电力业务系统的前端开发现状和需求进行调研分析, 基于目前已有的电力系统图形、GIS 地图、实时监控图和电力专业技术等进行组态化封装集成, 结合通用可视化页面元素组件, 整合实时库服务、商用库服务、业务应用微服务、大数据平台服务、文件服务等数据服务资源, 提供统一的接入方式和数据处理规范, 实现电力业务系统不同场景页面的快速开发和维护. 设计实现了一个适用于电力系统非生产大区的低代码可视化场景页面编辑引擎平台.

1 总体架构概述

面向电力系统的可视化页面编辑引擎 (以下简称页面编辑引擎) 面向电力系统前端业务, 旨在为开发者提供一个简单易用、界面优美、高效敏捷、智能化的页面编排和业务逻辑相结合实现的低代码页面开发平台. 通过本项目开发者以可视化的方式实现页面的布局、页面内容的编排、主题样式一键切换、人机交互业务逻辑处理、多源数据的动态绑定、页面一键发布预览、多技术前端框架的业务系统集成等, 快速开发出电力系统图形、信息化系统前端页面、可视化大屏页面、动态表单、智能报表、GIS 地图等满足电力业务系统的不同展示需求, 实现了敏捷开发和资源共享, 提高了前端开发效率, 从而解决人力资源浪费和系统维护不及时的问题.

1.1 系统总体结构设计

系统总体架构采用 B/S 架构实现, 浏览器端和服务器端采用前后端分离的开发模式, 实现前后端代码的解耦[3]. 从层级结构上设计为 4 个层次: 数据访问层、应用支撑层、可视化设计层和管理集成层. 系统总体架构如图 1 所示.

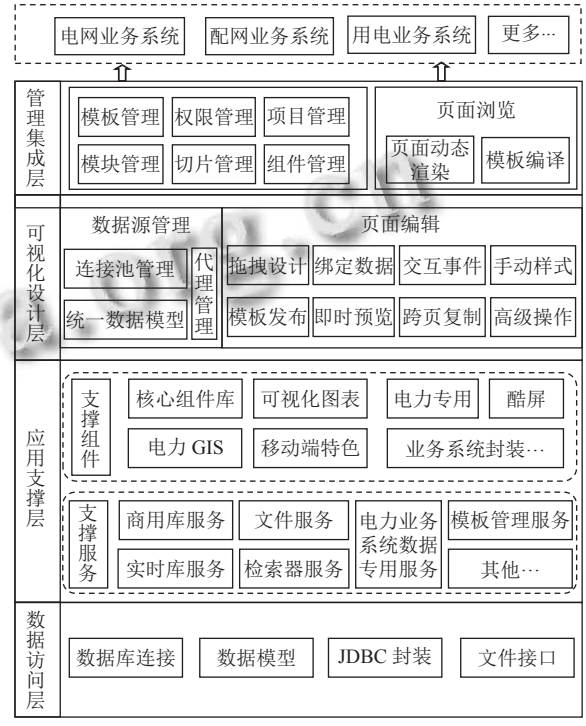

图 1 系统总体架构图

(1) 数据访问层负责对数据库和文件系统的读写操作, 为系统提供一个统一、安全和并发的数据持久机制, 并为系统支撑服务层提供数据服务.

(2) 应用支撑层主要描述支撑系统的各类组件库和支撑服务. 组件库为可视化页面设计提供基础组件支撑, 对于上层客户端的使用是透明的; 支撑服务通过接口的方式对前端提供数据服务, 根据业务系统的需求, 提供多种类型的服务接口.

(3) 可视化设计层即与用户交互的界面视图层, 是系统的核心子系统, 主要描述页面可视化编辑的各类功能, 主要包括构建页面动态数据的数据统一模型创建和数据处理服务、页面编辑流程的各个功能以及提供高可用性和易用性的人机交互操作的功能描述.

(4) 管理集成层主要描述业务系统项目、页面模板、模块、切片以及权限的管理操作处理流程, 提供与业务系统多种集成方式的页面模板, 根据业务系统的前端框架类型选择对应的集成处理方式. 对于同一业务系统的各页面模板, 可以选择模板管理统一对页面进行分级、分类、分模块管理.

1.2 浏览器端设计

MVVM (model-view-view model)[10,11] 是 MVP 模式[12] 与 WPF 结合的应用方式发展演变而来的一种新型架构框架, 是为了解决 MVC 模式控制层和视图层高耦合、未实现真正的分离和重用、视图无法组件化、代码复杂性高从而降低开发效率等问题而发展起来的. MVVM 模式将“数据模型双向绑定”的思想作为核心, 视图和模型之间通过绑定层进行交互, 模型和绑定层之间的交互是双向的, 因此视图和数据的修改会相互作用. MVVM 模式的低耦合性, 使得视图可独立于模型变化和修改, 为实现组件化提供了可能; 另一方面, 将视图逻辑放在绑定层中, 实现多视图共用一段视图逻辑, 开发人员只需要关注数据的变更即可, 提高了代码的可维护性.

传统的电力系统通常采用经典的 MVC 模式[13] 实现, 模型通过 RESTful 风格的接口、数据服务接口等方式与服务端通信, 当获取到数据后通过控制器通知视图进行更新显示内容[14], 但是随着业务不断地扩展, 这种模式的缺点越来越突出. 从本文的设计目标出发, 结合 3 种常用设计模式的优缺点, 系统前端采用 MVVM 设计模式, 有效的分离出视图层、视图数据绑定层、数据层以及功能逻辑层, 使各层之间处于高度解耦. 系统 MVVM 模式的示意图如图 2 所示.

1.3 系统功能设计

页面编辑引擎以组件库提供的组件为基础, 从后台服务获取数据为组件动态赋值, 通过可视化拖拽组件快速实现页面的搭建, 生成多种格式的页面模板为多种技术前端框架的业务系统提供无缝衔接式的集成. 系统各部分采用模块化设计的思想, 采用独立开发和发布的方式, 实现各模块之间的低耦合性, 并且在使用上保持生产独立性, 实现按需部署. 系统功能设计图如图 3 所示.

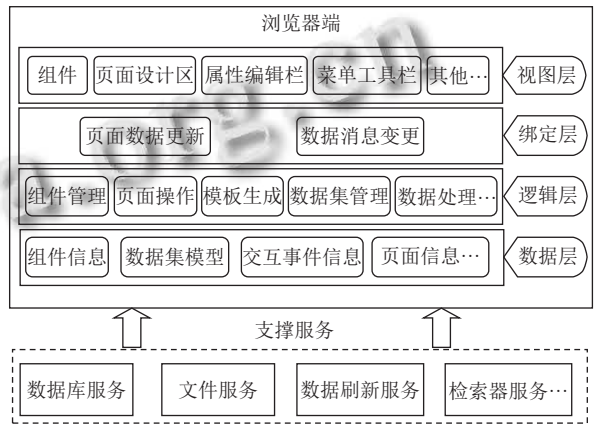

图 2 MVVM 模式示意图

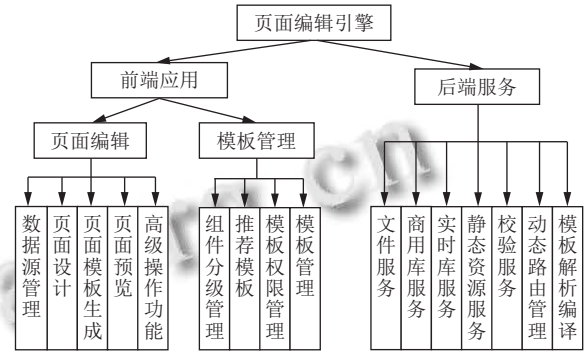

图 3 系统功能设计图

(1) 页面编辑是系统的核心, 提供业务系统前端页面的可视化编排以及业务逻辑的开发, 以拖拽方式实现可视化页面设计, 自动解析生成页面模板文件, 并根据业务系统集成需要生成多种模板格式, 实现一键发布预览.

(2) 模板管理提供对可视化页面编辑导出的页面模板进行分类管理、编辑、删除、权限控制等一系列管理操作. 并且提供给用户多种推荐布局模板, 快速构建页面整体布局. 通过提供页面集成多个模板作为页面的子模块, 快速完成页面的构建, 实现组装式页面开发.

(3) 后端服务提供前端交互接口以及业务逻辑处

理, 主要包括数据读写服务、文件读取与发布服务、路由管理、模板解析服务、校验服务等.

2 关键技术

针对目前电力系统前端页面手工编码开发方式效率低下、无法快速的响应用户需求的问题和现有可视化页面编辑的研究现状. 页面编辑引擎从 3 个方面进行着手研究和实现: 第一, 研究基于虚拟 DOM 的视图组件化页面可视化编辑技术, 实现页面快速构建和所见即所得的即时渲染效果, 分别从 UI 界面的角度和业务逻辑的角度进行分析, 将页面元素进行业务分割, 封装成组件, 实现资源复用和共享; 第二, 通过研究多源数据统一接入与动态分配技术, 实现不同来源的数据按照相同方式接入页面和多组件数据共享; 第三, 研究基于多类型模板的场景页面生成和解析技术, 将页面元素、后台数据和交互业务逻辑处理脚本进行规范性的整合生成场景页面描述模板, 实现页面的规范性和最大程度上的复用性. 根据业务系统的前端框架技术不同, 提供不同的页面模板格式, 实现与多种技术前端框架系统集成.

2.1 基于组态的可视化页面编辑技术

前端组件化[15] 的出现和基于组件的研发链路的广泛应用为页面可视化构建提供了最底层的保障, 降低了系统的耦合度. 构建可复用的组件库为可视化页面设计搭建提供基础的页面元素支撑, 页面的组件化划分有利于统一管理, 出现问题进行统一修改和替换[16]. 基于拖拽可复用组件实现可视化页面搭建的关键问题有: 组件的抽取和设计实现、组件的集成和动态数据的支撑、流畅的可视化拖拽效果支撑快速构建页面、所见即所得的即时渲染效果.

2.1.1 可复用组件的集成

页面编辑引擎从组件可扩展性和颗粒度[17] 角度设计实现, 按组件类型、业务需求范围把组件封装到多个组件库, 根据是否默认引入划分两类: 基础库和特色库.基础库包含页面通用的页面元素组件, 比如表单、表格等; 特色库包含大屏库、电力 GIS、业务系统自定义组件库等特色分类组件库. 为避免资源积累导致系统越来越庞大影响系统效率的弊端出现, 页面编辑引擎采用按需引入的方式集成所需类别的组件库. 其中基础库分类中的组件库是默认引入的, 组件库中所有组件对所有业务系统可见; 特色库分类中的组件库按照业务系统的需求由用户决定是否引入.

每个组件库在页面编辑引擎的资源区对应一个一级分类, 按照组件的类型再生成二级分类展示, 组件库中的组件在所属分类下生成可拖拽的组件快照. 页面编辑引擎对集成进来的每个组件提供一个专属的属性资源文件, 记录组件的快照基础信息、属性编辑信息、交互事件信息和渲染方法. 当拖放一个组件到可视化设计区域时, 自动读取对应的属性资源文件, 在配置区域显示组件的基础信息、可编辑的属性信息、动画效果配置信息和交互事件信息供用户进行编辑, 实现组件的个性化效果.

2.1.2 基于 HTML5 的可视化组件拖放

在 HTML5 之前, 实现 Web 对象的拖放操作需要依靠 mousedown、mousemove 等鼠标响应事件、编写大量的 JS 代码实现[18], 因此 Web 中实现这个功能操作比较复杂. HTML5 标准中引入了直接支持拖放操作的 API, 大大简化了网页元素的拖放操作难度和复杂度. 本文采用 HTML5 拖放技术实现通过拖拽组件快速构建场景页面, 实现步骤如下:

(1) 组件快照选中

1) 组件快照设置 draggable 属性; 2) 实现 dragstart 事件: 设置当前元素为选中元素; 3) 实现 dragend 事件: 获取并设置当前元素为空.

(2) 组件快照拖动

1) 实现 dragenter 事件: 获取当前元素, 判断途径区域是否允许拖放, 根据判断结果更新拖拽时的 CSS 样式; 2) 实现 dragover 事件: 判断目标元素是否允许当前元素作为子元素, 允许标记为有效拖放, 不允许标记为无效拖放; 3) 实现 dragleave 事件: 对当前元素移除可拖放的 CSS 样式.

(3) 拖拽完成释放

实现 drop 事件: 首先获取拖拽元素和目标元素的信息, 根据操作类型执行不同的流程操作, 然后构建组件以及相关联组件信息, 然后构建新的组件树并记录此次操作为撤销操作缓存数据, 最后删除拖拽标记并发布更新页面的通知. 组件释放执行的操作流程如图 4 所示.

2.1.3 基于虚拟 DOM 的页面渲染技术

可视化页面构建平台在搭建页面的过程中, 通过不断的拖放组件并修改其属性达到页面展示效果, 就会频繁的执行 DOM 操作, 从而导致页面重绘和回流,

这种跨界交流是高成本的, 会造成页面渲染缓慢. 虚拟 DOM[19] 本质上是一种模拟 DOM 树的 JS 树形数据结构, 每个节点对应一个 JS 对象, DOM 操作先在虚拟 DOM 上执行对比, 然后有针对性的更新差异化部分, 实现页面的局部刷新. 另一方面, 在响应式需求下, 通过 JS 直接操作 DOM 会造成视图和模型不匹配, 当有状态变更时构建的界面会重新渲染整个视图, 此时就需要修改整个 innerHTML, 当数据量变化很小时会浪费大量的资源, 造成性能下降. 电力系统可视化编辑构建场景页面是一个逐步累计添加、完善的操作, 因此使用虚拟 DOM 技术可以实现所见即所得的页面即时预览效果.虚拟 DOM 算法包括如下步骤:

(1) 用 JavaScript 对象结构模拟真实的 DOM 树;

(2) 当有状态变更时, 新生成一个对象结构, 比对两棵虚拟 DOM 树的差异;

(3) 将差异应用到真实 DOM 树.

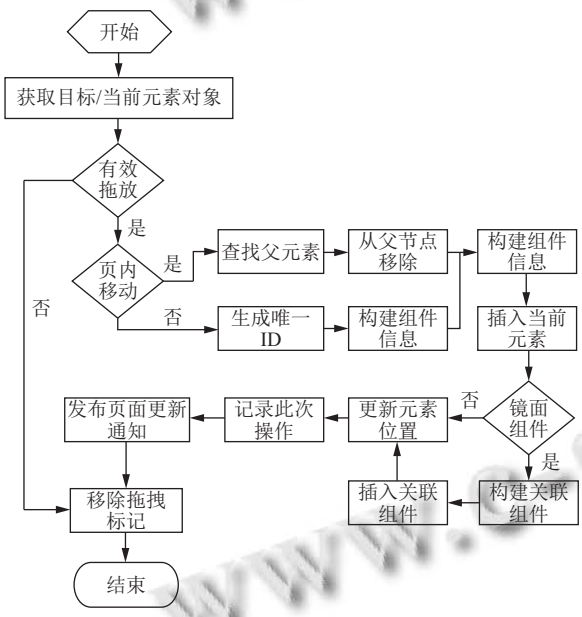

图 4 组件释放执行操作流程图

比对差异算法 (Diff) 是虚拟 DOM 算法的核心部分. 传统的 Diff 算法[20] 通过树深度优先搜索算法循环递归对节点逐个进行对比, 时间复杂度为 $O(n^3)$, 考虑到页面构建过程中组件的层次关系和属性修改的频繁性, 一个不太复杂的页面组件一般也会达到 30 个以上, 这种指数级别的性能消耗代价太高, 不能满足页面可视化编辑引擎的页面渲染需求. 基于传统 Diff 算法的不足和 React 开发者对于 DOM 树策略做出的假设, 算法有了如下优化改进: 对虚拟 DOM 树按照层级控制, 只比较同层级的节点, 当节点被删除时, 节点以及子节点不再进行递归比较; 当节点类型相同时, 更新节点的属性; 当发现新旧两个节点类型不同时, 直接删除旧的节点及其子节点, 并插入新的节点; 对于在一个层级下的同组子节点如果仅仅是位置不同, 进行移动操作.

页面编辑引擎可视化构建页面时, 涉及 4 个方面的操作: 插入新的组件、修改组件属性、移动组件位置、删除组件. 差异化算法流程步骤如下:

(1) 在页面设计区域内, 构建新旧两颗虚拟 DOM 树结构;

(2) 采用深度优先搜索遍历两棵 DOM 树, 从根节点开始进行比较操作;

(3) 判断新节点类型是否为空, 如果为空, 标记删除这个节点;

(4) 判断新旧节点是否为同一类型的节点;

(5) 如果节点类型相同, 则判断引用是否相同, 如果引用相同表示未更新, 则返回, 继续子节点的遍历;

(6) 如果引用不同, 判断是否文本节点, 如果是文本节点, 比较文本节点内容, 标记更新文本节点内容;

(7) 如果不是文本节点, 比对节点属性, 如果存在差异性, 标记更新节点属性, 然后判断比对子节点;

(8) 如果新旧节点都拥有子节点, 并且子节点不相同, 进入子节点比对流程;

(9) 如果只有新节点有子节点, 标记添加此子节点;

(10) 如果只有旧节点有子节点, 标记删除子节点操作;

(11) 如果节点类型不同, 标记删除旧节点, 插入新节点操作.

Diff 算法在逐层比较时会保存标记的对比差异, 比对结束后, 通过递归迭代遍历这个保存的差异对象, 对有差异的节点执行相应的 DOM 操作, 实现有针对性的局部更新.

虚拟 DOM 技术的本质是用 JS 的运算性能消耗来换取跨界操作 DOM 的性能消耗, 在页面模板生成之后集成到系统中运行时, 在大部分场景下页面的数据更新量比较少, 在虚拟 DOM 上进行合理的优化能够获得很好的渲染性能. 但是在一些实时性、高并发等性能要求极高的应用中, 采用虚拟 DOM 需要不断地进行数据对比, 而 DOM 操作量并没有减少, 此时额外消耗大量的虚拟节点对比时间, 使得实际性能更为低

下[21]. 此种情况下, 提倡开发者在构建页面的过程中, 放弃系统采用的默认方式, 在系统提供的用户自定义业务逻辑处理脚本中操作实时数据的真实 DOM 完成数据变更, 并进行合理的 DOM 优化, 尽量合并页面中频繁变更的数据 DOM 操作, 并采用请求限流和缓存等优化方案提高页面的渲染性能.

2.2 数据统一接入和动态分配技术

电力系统在信息化领域的不断深入和发展, 产生了大量的异构数据源, 页面编辑引擎作为一个软件平台应能够轻松整合多源异构数据源, 通过统一的数据模型接入, 运用可视化技术实现数据模型[22] 可视化、流程可视化, 以更直观、更简洁的方式实现数据的整合. 本文提出的基于统一数据模型和动态分配技术的可视化组件动态数据逻辑如图 5 所示.

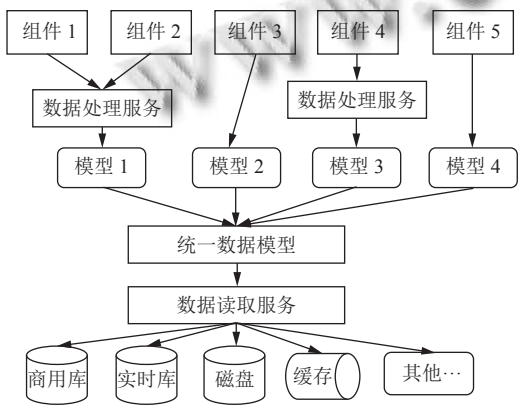

图 5 组件动态数据逻辑图

页面编辑引擎创建 JSON 结构的统一数据模型称为数据集, 实现异构来源数据的有效统一接入和数据访问接口的应用. JSON 结构中包含唯一标识 ID、名称、数据服务类型、服务地址、服务请求方式、默认参数、当前参数、数据转换方法、备注信息. 根据数据来源的不同调用不同的支撑服务来获取动态数据.

页面中的元素是一个个的组件, 组件通过绑定数据集的方式接入动态数据. 数据集和页面中的组件是一对多的关系. 如果多个组件绑定了同一个数据集, 但每个组件接收的数据格式不同, 或者一个数据集仅被一个组件绑定, 但是其服务接口返回的数据格式和组件所需数据格式不一致, 那么将采用数据处理服务进行智能的转换.

数据动态分配技术采用发布-订阅模式实现数据的自动化读取和组件的动态数据刷新渲染.当页面在浏览器中打开时, 统一数据模型中所有绑定组件的数据集自动请求相应的数据服务, 将获取到的动态数据通过数据处理服务处理后赋值给绑定的组件, 并根据用户需求决定是否将获取到的数据存储到统一数据模型的相同结构的数据结果集中. 组件订阅了数据集的变化, 当收到数据变动的通知, 从而更新视图, 实现组件动态数据的实时渲染.

2.3 基于模板的场景页面生成技术

传统手工编码方式实现电力系统的一个前端页面通常是在搭建好的框架中编写 HTML 页面, 并引入依赖的 JS 文件和 CSS 样式, 编写相应的业务逻辑处理操作, 考虑到每个开发者的编码习惯和技术水平存在一定的差异性, 不可避免的产生很多重复的工作. 另外一个方面, 由于每个业务系统由不同的团队搭建和开发, 采用的技术和框架也不相同, 这就给后期的维护带来了极大的困难. 因此页面编辑引擎基于以上两个方面考虑, 本文提出基于多类型模板的页面编辑思想, 使用户不必考虑技术水平的差异性, 实现页面的规范性和最大程度上的复用性. 根据业务系统的前端框架技术不同, 提供不同的页面模板格式, 实现与多种技术前端框架系统的集成.

电力系统一个场景页面通常包含完整的业务功能, 因此页面模板应包含完整的业务逻辑, 有助于快速生成业务页面, 不同的页面模板适用于不同的业务功能. 从已有模板中选择合适的页面模板并派生出默认业务页面, 再对默认页面进行可视化编辑, 从而生成目标业务页面, 这是一个继承式的资源积累过程. 基于模板的场景页面生成技术包含 2 个方面的核心内容: 模板生成引擎和模板解析引擎.

模板生成引擎将页面元素、后台数据和交互业务逻辑处理脚本进行规范性的整合生成场景页面的描述模板, 根据业务系统的前端框架技术不同, 提供不同的页面模板格式, 与多种技术前端框架系统无缝衔接. 对于一个场景页面来说, 通过拖拽组件的方式快速搭建的页面结构就是一个场景组件树, 场景组件树定义了组件间的父子兄弟层级关系, 树中的节点就是页面组件. 因此, 模板生成引擎的本质就是解析场景组件树、处理各层级组件数据流以及自定义业务逻辑交互脚本, 按照规范的格式构建页面模板语言的过程. 模板生成引擎流程如图 6 所示.

场景组件树解析的过程就是遍历树, 解析树的每

个节点的内容并构建模板视图的一个过程, 树遍历算法分为深度优先搜索 (DFS) 和广度优先搜索 (BFS), 根据场景组件树的结构特点, 结合两种搜索算法的最优适用场景, 本文选择深度优先搜索算法[23] 实现. 深度优先搜索算法根据组件之间的父子链式关系, 找出组件包含的所有子组件, 解析每个节点的信息, 结合数据信息、参数信息和样式信息等构建场景页面的模板. 场景页面的开发往往是一个根据现场需求不断变更、持续迭代的过程, 不可能一次就开发完成, 也不可能一成不变, 页面的二次编辑就是对页面组件的重新组合, 以及编辑各组件和页面的内容. 模板解析引擎的原理是将模板语言转换成可视化页面编辑引擎识别的结构, 重新构建场景组件树和组件绑定的数据集结构树, 并识别出用户自定义的人机交互业务逻辑脚本和自定义的样式, 实现页面模板的重新可视化编辑.模板解析流程是模板生成流程的逆向操作, 能够自动根据语法结构进入相应的解析流程.

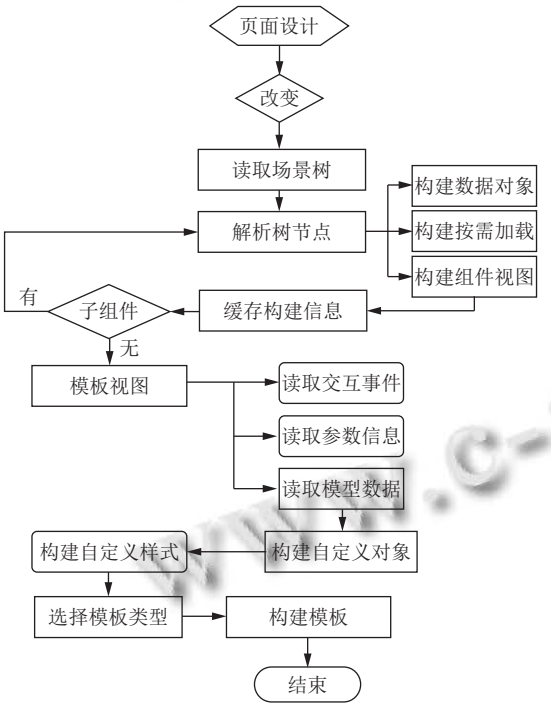

图 6 页面模板生成流程图

3 应用实例

本文设计实现的面向电力系统的基于组态的可视化页面编辑引擎目前已成功应用于调控云系统运行辅助分析模块, 开发的功能已经部署在华东现场投入使用; 同时在电力现货市场支持系统中也得到很好的推广应用, 已经在内蒙古、甘肃、江苏等现场部署运行; 国家能源集团生产运营协同调度系统一期项目基础支撑平台、智能调度、在线监测等模块的部分页面也使用此引擎平台开发完成, 目前项目已经在北京现场正式部署运行, 并顺利通过国能集团的验收工作, 页面展示效果和性能均满足现场的展示需求.

现有的电力业务系统前端技术路线和搭建的前端框架种类不尽相同, 开发方式主要依赖于前端专业开发人员手工编码实现, 本文提出的基于组态的可视化页面编辑引擎大大地降低了开发门槛, 提高了开发效率, 和传统编码方式对比如表 1 所示.

表 1 页面编辑引擎和传统编码方式对比评估

评价指标		传统方式	可视化编辑
技术要求		熟悉业务系统使用的框架和技术、熟悉前后端交互、熟悉性能优化方案的专业开发者	无特殊要求, 了解基础JS写法, 能编写业务处理脚本
平均耗时	简单	1–3天	<1天
	中等	3–5天	<3天
	复杂	5–10天	<5天
性能		依赖于开发者水平	大部分场景下有较高的性能
灵活性		灵活度高, 编码者自由发挥	需要遵循一定的编写规范, 有一定的学习成本
可维护性		依赖于开发者编写规范和对当前业务熟悉程度, 改动复杂	可视化编辑, 可维护性强, 易于修改
通用性		对框架技术依赖性强, 通用性低	支持多种导出类型, 适用于目前大部分主流前端框架搭建的系统
规范性		依赖于开发者水平	提供统一模板格式

在本地局域网测试环境下, 使用本文提出的系统开发的页面, 在不考虑后端返回数据影响耗时的情况下, 并发 300 个用户 80% 的页面 5 s 内完成页面渲染, 单用户请求下, 90% 的页面 2 s 内完成页面渲染. 在国分调控云运行环境下, 对相同复杂度的页面进行测试, 基于 DOM 驱动的模板引擎和本文基于虚拟 DOM 开发的页面性能对比结果如表 2 所示.

表 2 中测试结果数据受测试环境网络影响, 页面总体渲染时间比较长, 原因是测试环境为远程访问, 页面的资源请求和后端服务返回数据耗时比较长. 从测试结果可以看出, 页面采用虚拟 DOM 技术总体性能可以满足现场需求, 在页面局部更新数据时更能体现其优势.

4 结论与展望

实践证明, 面向电力系统的页面编辑引擎实现了电力系统前端页面的基本组成单元的原始积累, 随着业务系统的推广使用, 组件物料积累越来越丰富, 开发效率也将呈现上升趋势. 系统提供一体化页面模板在线发布、动态路由生成和页面热加载, 实现页面的即时发布与在线预览. 避免用户手动操作上传模板、创建路由、重启项目等操作, 提升了高级应用能力, 展示了页面编辑引擎的自动化管理能力. 系统实现了场景页面的可视化构建, 在更高级层面上对用户屏蔽页面编写细节, 可应对电力系统不同场景下页面的低代码敏捷高效开发需求, 为推动电力系统发展和新一代电力系统的构建提供基础支撑和强有力的保障. 为进一步满足用户对于系统操作人性化和易用性的需求, 下一步工作将加强对已有系统的自动化、流程化和精细化管理, 对跨平台、跨端的兼容性支持研究.

表 2 页面性能对比结果 (ms)

页面操作		操作DOM		虚拟DOM	
		总时间	加载渲染时间	总时间	加载渲染时间
简单查询	第一次加载	1 190	231	1 240	248
	缓存刷新	970	145	1 100	77
	局部刷新	920	126	489	32
较复杂图表	第一次加载	4 247	322	2 674	253
	缓存刷新	4 056	305	2 376	237
	局部刷新	3 966	286	2 436	122
复杂图表和列表	第一次加载	7 976	525	3 948	476
	缓存刷新	5 230	434	3 560	273
	局部刷新	3 280	416	1 152	187

参考文献

1 薛芸, 廉东本, 王俊霖. IDC 可视化软件开发平台中的配置与编译子系统. 计算机系统应用, 2017, 26(3): 280–283. [doi: 10.15888/j.cnki.csa.005650]

2 宋奕爽, 刘绍华. Web 端可视化表单生成引擎的设计与实现. 软件, 2017, 38(12): 153–159. [doi: 10.3969/j.issn.1003-6970.2017.12.029]

3 李隆隆. 可视化页面编辑平台的设计与实现 [硕士学位论文]. 南京: 南京大学, 2018.

4 江婷, 林嘉琦, 马建雄, 等. 基于组件化的数据可视化系统设计与实现. 智能物联技术, 2019, 51(1): 48–54.

5 严亚勤, 邱健, 訾鹏. 一种基于 Web 的电网图示化编辑器的设计与实现. 电力系统保护与控制, 2010, 38(8): 92–96. [doi: 10.3969/j.issn.1674-3415.2010.08.020]

6 杨德尚. 电力系统可视化平台软件开发与研究. 2009 电力行业信息化年会论文集. 北京: 中国电机工程学会, 2009: 37–40.

7 赵瑞. 面向电力系统的 SVG 编辑器的设计与实现 [硕士学位论文]. 北京: 华北电力大学, 2012.

8 赵剑波, 王蕾. “十四五”构建以新能源为主体的新型电力系统. 中国能源, 2021, 43(5): 17–21. [doi: 10.3969/j.issn.1003-2355.2021.05.003]

9 曾鸣, 杨雍琦, 李源非, 等. 能源互联网背景下新能源电力系统运营模式及关键技术初探. 中国电机工程学报, 2016, 36(3): 681–691.

10 何焕春, 杨怿. 基于 MVVM 构架的 Web 前端框架研究. 电脑知识与技术, 2017, 13(24): 59–60.

11 王宇. 基于 MVVM 前后端分离的物联网维管系统的研究与实现 [硕士学位论文]. 北京: 北京工业大学, 2019.

12 易曼萍. 基于 MVP 架构的任意波形发生器人机接口设计 [硕士学位论文]. 成都: 电子科技大学, 2016.

13 贾顺贺, 陈建飞, 陈古运, 等. 基于 MVC 架构的个人健康信息管理系统设计与实现. 计算机应用与软件, 2018, 35(3): 43–48. [doi: 10.3969/j.issn.1000-386x.2018.03.008]

14 黄昆, 赵昆, 杨立波, 等. 电网调控系统轻量化人机交互体系架构及关键技术. 电力系统自动化, 2019, 43(7): 159–165. [doi: 10.7500/AEPS20180525001]

15 王萌, 田杨, 李宁宁. 组件化 WEB 前端架构设计与实现. 电脑知识与技术, 2018, 14(30): 77–79.

16 周伟, 郑世珏. Web 前端工程化解决方案研究. 信息技术, 2018, (8): 44–47.

17 李春江, 张鹏, 李江, 等. 基于 Vue 的业务中台组件设计与实现. 科学与信息化, 2020, (36): 49.

18 刘耀钦. 利用 HTML5 拖放技术实现多文件异步上传. 四川理工学院学报 (自然科学版), 2015, 28(1): 17–20, 30.

19 戴志诚, 程劲草. 基于虚拟 DOM 的 Web 前端性能优化研究. 计算机应用与软件, 2017, 34(12): 21–25, 31. [doi: 10.3969/j.issn.1000-386x.2017.12.004]

20 张贺峰, 特日根. 基于虚拟 DOM 的空间数据列表渲染方法研究与实现. 测绘与空间地理信息, 2021, 44(6): 19–22, 26. [doi: 10.3969/j.issn.1672-5867.2021.06.006]

21 顾睿. 基于 MVVM 的虚拟 DOM 轻型 Web 前端框架的设计与实现 [硕士学位论文]. 西安: 西安电子科技大学, 2019.

22 宋美娜, 崔丹阳, 鄂海红, 等. 一种通用的数据可视化模型设计与实现. 计算机应用与软件, 2017, 34(9): 38–42, 96. [doi: 10.3969/j.issn.1000-386x.2017.09.008]

23 陈锋. 基于 DFS 图的遍历路径优化分析. 电脑与信息技术, 2021, 29(1): 4–5. [doi: 10.3969/j.issn.1005-1228.2021.01.002]

(校对责编: 孙君艳)