ExtJS-JSP-Vue.js 3 Hybrid Architecture: A Case Study in Enterprise Web Application Development

Mingchen Qiu

Nanyin Faba Consumer Finance Co., Ltd., Nanjing 210000, Jiangsu, China

Abstract: ***To address the ecosystem loss, technical debt, and talent gap caused by the discontinuation of EXTJS updates in enterprise applications, this study accumulates a low-cost, gradual evolution and upgrade iteration approach for enterprise applications. Focusing on enterprise-level Web front-end visualization integration, it proposes a configuration method that enables EXTJS to dynamically load Vue3 components in conjunction with JSP to build front-end user interfaces, offering the following advantages: (1) Compared with traditional iframe embedding, the configuration method described in this paper has superior loading efficiency, improving user access speed and reducing cross-framework state synchronization latency ≤ 50ms. (2) In high-interaction modules, such as data visualization requiring roll-up and drill-down, pages using Vue3 dynamic loading can achieve real-time interaction and animation effects. (3) In terms of development cost, it saves 70% compared with full refactoring, and the upgrade cycle is shortened to one-third of the full refactoring plan, significantly reducing the cost and time investment of enterprise technology upgrades. (4) The architecture described in this paper is highly flexible, supporting dynamic switching of component loading strategies, and enabling or disabling Vue modules according to different business scenarios. Thus, this technology has broad application prospects.***

Keywords: JSP; Vue3; EXTJS Integration Introduction.

1. INTRODUCTION

EXTJS, commonly abbreviated as Ext, was released in 2006 as a feature-rich JavaScript framework. Built on pure HTML/CSS + JS, it is mainly used for creating front-end user interfaces. Thanks to its rich UI elements, excellent cross-browser compatibility, object-oriented and extensible design, high responsiveness, and abundant documentation and examples, EXTJS combined with JSP once dominated enterprise UI construction, making some applications dependent on it. In 2023, EXTJS ceased maintenance, dealing a severe blow to the large number of legacy systems built with EXTJS and JSP, exposing them to the risk of technological obsolescence. Yet rebuilding with new frameworks would not only disrupt business continuity but also impose financial pressure on the company. Against this backdrop, a method is needed that can keep existing EXTJS+JSP applications running stably, seamlessly integrate the strengths of modern front-end components such as Vue3, and also detach sub-applications that meet current UI paradigms from the legacy system for architectural decoupling. To address these challenges, this paper proposes a micro-front-end-based progressive refactoring plan: by constructing a hybrid EXTJS and Vue3 architecture, the core functions of the legacy system are preserved while modern components are seamlessly integrated via the Qiankun micro-front-end framework. The solution focuses on three key technical layers: first, designing a sandbox isolation mechanism to resolve cross-framework conflicts; second, developing a standardized bridging layer to enable EXTJS to dynamically load Vue3 sub-applications; and finally, refactoring the build and deployment pipeline. Practice shows that this approach effectively reduces migration costs, equips legacy systems with modular extensibility, and offers a viable path for the technological evolution of enterprise applications. Lin (2025) establishes an enterprise AI governance framework that balances innovation and risk through product management principles [1], while healthcare diagnostics are advanced by Wang (2025) with RAGNet, a transformer-GNN-enhanced model for rheumatoid arthritis risk prediction [2]. Energy systems research includes Gao, Tayal, and Gorinevsky (2019), who developed probabilistic planning for minigrids with renewables [3], and Gao and Gorinevsky (2020), who optimized resource mix using probabilistic modeling [4]. Network traffic analysis is enhanced by Zhang et al. (2025) through MamNet for time-series forecasting [5], while computer vision sees multiple contributions from Peng et al. with 3D Vision-Language Gaussian Splatting [6], domain-adaptive human pose estimation [7], and NavigScene for autonomous driving navigation [8]. Supply chain optimization is addressed by Tang, Yu, and Liu (2025) through dynamic pricing models [9], while robotics research includes Guo (2025) on optimal trajectory control [10] and Guo and Tao (2025) on robot-environment interaction modeling [11]. Software architecture advances include Zhou (2025) on microservices performance optimization [12], and data security is strengthened by Zhang (2025) through blockchain-based medical data sharing [13]. Market analysis capabilities are expanded by Yu (2025) using Python applications [14], while healthcare delivery is transformed by Wei et al. (2025) with AI-driven telemedicine systems [15]. Cross-media analytics are advanced by Yuan and Xue (2025) through data fusion frameworks [16], and computer vision applications include Chen et al. (2022) with gaze-estimated object referring [17]. Industrial applications feature Tan et al. (2024) with transfer learning for fault diagnosis [18], while digital marketing strategies are theorized by Zhuang (2025) for real estate [19]. Recommendation systems are enhanced by Han and Dou (2025) through hierarchical graph attention networks [20], and sales forecasting is improved by Zhang, Jingbo et al. (2025) in gaming industry applications [21]. Business automation sees multiple contributions from Zhu (2025) with TaskComm for workflow optimization [22] and ReliBridge for platform stability [23], while 3D content creation is advanced by Hu (2025) with few-shot neural editors [24] and low-cost authoring pipelines [25]. Manufacturing optimization is addressed by Xie and Chen (2025) with multi-agent task recognition systems [26] and business intelligence through CoreViz [27]. System reliability engineering includes Zhu (2025) with RAID for ad systems automation [28] and Zhang, Yuhan (2025) with SafeServe for release safety [29]. Advertising technology is transformed by Hu (2025) with UnrealAdBlend for immersive content [30], while privacy preservation is ensured by Li, Lin, and Zhang (2025) through federated learning frameworks [31]. Sequential recommendation is enhanced by Li, Wang, and Lin (2025) using graph neural networks [32]. Urban planning applications include Xu (2025) with CivicMorph for public space development [33], while network testing is automated by Tu (2025) with AutoNetTest for 5G diagnostics [34]. Finally, causal recommendation systems are advanced by Wang (2025) through joint training for MNAR data [35].

2. OVERALL SOLUTION DESIGN

When transforming enterprise applications, business imposes two demands on technology: (1) cost; (2) business continuity. Therefore, the architecture must meet the following requirements: (1) achieve decoupling of front and back ends; (2) enable seamless integration of old and new technology stacks; (3) introduce modern capabilities while ensuring the stability of legacy business.

Table 1 shows the tech stack and responsibility boundaries of the main application and sub-applications. It can be seen that the configuration method described in this paper continues to use the EXTJS-plus-JSP rendering main application to carry existing business logic such as lists, routing, lists, and form submission, while the Vue3 sub-application is mainly used to support highly complex interactive modules. In addition, a configuration-driven approach is adopted to achieve separate deployment of front-end resources (Vue components) and back-end services (JSP), implementing partial front-end/back-end separation in micro-scenarios and reducing the front-end's dependence on the JSP rendering model, thus forming a configuration-driven “dual-track” hybrid architecture.

Table 1: Tech Stack and Responsibility Boundaries of Main Application and Sub-applications

应用类型	技术栈	职责边界
主应用	EXTJS + JSP	路由管理、基础表单、列表展示等
子应用	Vue3 + Webpack	数据可视化穿透、高级的筛选器等

3. KEY TECHNOLOGIES

The key technologies of the configuration method described in this paper are as follows. 1) Dynamic component loading engine: leveraging modern front-end frameworks' code-splitting capability to load Vue3 and other modern framework components on demand, precisely matching the dynamic tech-stack requirements of different business scenarios. 2) Cross-framework context communication: relying on deep integration between the micro-front-end engine and the EXTJS-JSP architecture, it establishes a state-sync mechanism between EXTJS and Vue3, supporting parameter passing, data linkage, and other core functions to ensure efficient and accurate cross-framework data exchange. 3) Sandbox compilation environment: through a customized Webpack build chain, Vue components are independently compiled and isolated within the EXTJS-JSP system, effectively avoiding logic and style conflicts when multiple frameworks coexist. 4) Request middleware: using the Nginx static server for resource proxying and cross-origin handling, it breaks the same-origin policy limitation of the B/S architecture, ensuring seamless integration and stable loading of runtime resources.

4. CONFIGURATION STEPS

4.1 Plan the "hybrid" Project Directory Structure and Configure the Strategy for EXTJS + JSP to Access the Vue3 Directory.

Transform the traditional Java-web project into a hybrid architecture of "Java back-end + partial micro-front-end".

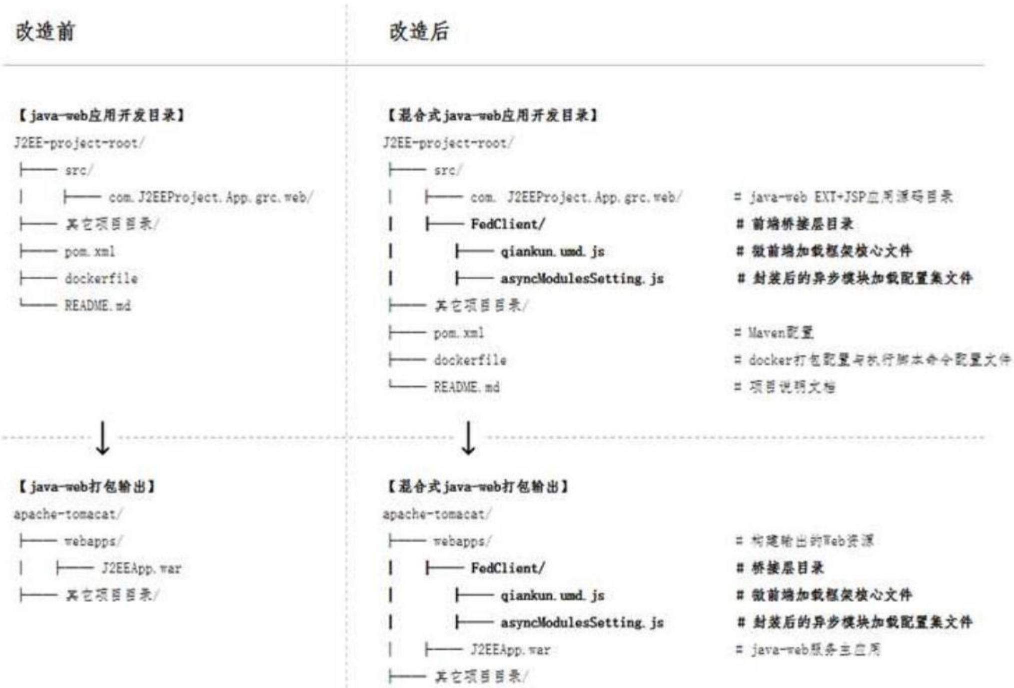

Figure 1: "Hybrid" Java-web project directory structure

Figure 1 shows the pre-migration architecture. From Table 1 we can see that the pre-migration structure consists of Java back-end source code together with basic configurations such as Maven and Docker. After migration, a "hybrid" project structure is formed, the key being the placement of a front-end bridging layer within the Java back-end project directory.

Specifically, both the source and the compiled-output directories now include a FedClient/ directory (the front-end bridging layer), which contains the core files of the micro-front-end loading framework and the asynchronous module loading configuration files. The original Java back-end code and Maven configuration remain unchanged. When the Java-web project is built, the resulting WAR package automatically includes the contents of the webapp/ directory, so the front-end bridging layer FedClient/ directory becomes accessible after Tomcat maps the resources (e.g., http://localhost:8080/FedClient/).

In other words, both the source code and the compiled output now contain a new FedClient/ directory (the front-end bridging layer); it embeds the core files of the micro-frontend loading framework and the asynchronous module-loading configuration, while the original Java back-end code and Maven configuration remain unchanged. The WAR package produced by the Java-web build automatically includes everything under webapp/, so after Tomcat maps the resources, the front-end bridging layer FedClient/ directory becomes accessible.

The configuration described in this article adds a stand-alone Vue3 front-end application project directory that can be developed and deployed independently, used to develop, build, and package Vue3 components that EXYJS can load remotely.

The final deliverable of this phase not only gains modern front-end capabilities—flexible module loading, independent development, and independent deployment—but also retains the mature Java back-end ecosystem, giving the project more flexible front-end/back-end collaboration. From an architectural-extensibility perspective, the modern SPA front-end framework hosted in the sub-application can, if needed, be swapped out for a non-Vue3 framework (e.g., React or Angular).

4.2 Refactor the EXTJS + JSP Project and Introduce a “bridging layer” to Configure Cross-framework Communication, Enabling Hybrid Rendering. Table 2 Lists the Key Points for Micro-frontend Integration of the EXTJS-JSP Main Interface Entry File.

Table 2: Key points for micro-frontend integration of the EXTJS-JSP main interface entry file

改造 EXTJS 联合 JSP 工程源码 - 主界面入口 jsp 文件	
引入微前端引擎文件,作为集成 VUE3 应用的桥接层核心。	<script src=" /FedClient/qiankun.umd.min.js"></script>
封装后的异步模块加载配置集文件作为加载 Vue3 应用的配置驱动核心。	<script src="/FedClient/asyncModulesSetting.js"></script>
HTML 中布置 Vue3 应用的 DOM 容器作为预加载 Vue3 资源的初始化。	<div data-controller=" app_FedClient"> 插槽 </div>
在入口 jsp 文件 标签外的 javascript 环境中进行预加载的调用。其 中”asyncExt“是自定义的全局对象,作为桥阶层核心管理对象。	asyncExt.useFedClient(‘ ’).then(instance => {console.log(为预加载成功 ’); }, subExport => { console.log(‘重复创建 : ’, subExport); });

Import the micro-frontend engine “qiankun” via the UMD specification, and use the loadMicroApp mode to load the Vue3 sub-application on demand. Re-encapsulate the full micro-scenario lifecycle so that Promise-based configuration manages the Vue3 asynchronous loading lifecycle, improving code readability. Control container creation and placement via a custom DOM attribute “data-controller”. The most critical part is implementing, in asyncModulesSetting.js, a global “bridging layer” static class “asyncExt” that can be configured to drive EXTJS + JSP to dynamically load Vue3 components. Table 3 shows the bridging-layer module design for loading Vue3 sub-applications in the EXTJS-JSP project.

Table 3: Bridging-layer module design for loading Vue3 sub-applications in the EXTJS-JSP project

布置 EXTJS 联合 JSP 工程加载 Vue3 子应用的桥接层 - asyncModulesSetting.js 文件	
A: 桥阶层单例工 厂创建模块	asyncExt.module(‘FedClientFactory ’, options => { ... });
B: 桥阶层初始化 配置模块	asyncExt.module(’MicroModuleSetConfig ’ , (microModule, props) => { ... })
C:Vue3 组件加载 配置模块	asyncExt.module(‘useFedClient’, options => {... })

To make configuration-driven loading work, register three core modules via the asyncExt.module method provided by asyncModulesSetting.js. Use them in the ExtJS context: A: the bridging-layer singleton factory creates a singleton “bridging-layer” object; B: after the bridging-layer initialization configuration module is set up, it can be obtained anywhere; C: the Vue3 component loading configuration module is invoked on demand when ExtJS opens a route or panel, dynamically fetching Vue3 components to co-construct the UI.

4.3 Refactoring the Vue3 Build Tool Webpack Environment Configuration

In the development environment configuration [H2.1], set up the static assets directory via Webpack and enable Gzip compression, launch a local service on port 9110, and turn on hot-reload; simultaneously configure a proxy to forward requests on the /JavaWeb path to the Tomcat service (port 8063), resolving CORS by modifying the Host header and adding the Access-Control-Allow-Origin: * response header. To avoid cross-framework conflicts [H1.1], adopt a sandboxed build-chain configuration: JS files are transpiled by Babel with node_modules dependencies excluded, CSS files use modular processing with isolated class names generated in the [local]_____[hash:base64:5] hash format, and a _SANDBOX_ global flag is injected via DefinePlugin. In the production build configuration [H3.1], set the public asset path to /FedClient/dist/, append 8-character full hashes to JS/CSS filenames, and use the [name].[hash]-[ext] pattern for images; define the sub-application name as "Export", output it as a UMD module, and declare the webpackJsonp_Export global variable for ExtJS to invoke.

4.4 Nginx CORS Solution

Configure the Nginx service [H4.1] to listen on port 9120 and bind to the localhost domain, establishing a dual-path proxy: route the root path / to the J2EE application service (port 8080) while proxying /FedClient/dist/ to the Vue3 application resource location; disable proxy redirect response headers and set custom error pages for 50x errors. Ultimately, unified access to the front-end interface—where ExtJS dynamically loads Vue3 components—is achieved via http://127.0.0.1:9120/, completely solving the cross-origin issue.

5. CONCLUSION:

Addressing the technological discontinuity risk faced by enterprise web applications after ExtJS maintenance ceased, this study proposes an innovative hybrid architecture that seamlessly integrates the traditional ExtJS-JSP stack with the modern Vue3 framework. Through key technologies such as a dynamic component-loading engine, cross-framework communication mechanism, and sandboxed build environment, a configuration-driven “dual-track” hybrid architecture is successfully constructed, preserving the business stability of legacy systems while significantly enhancing the user experience of highly interactive modules and cutting full-scale refactoring costs by 70%. Thus, the architecture presented herein has demonstrable value for wider adoption.

REFERENCES

[1] Lin, T. (2025). Enterprise AI governance frameworks: A product management approach to balancing innovation and risk. International Research Journal of Management, Engineering, Technology, and Science, 1(1), 123–145. https://doi.org/10.56726/IRJMETS67008

[2] Wang, Y. (2025). RAGNet: Transformer-GNN-Enhanced Cox–Logistic Hybrid Model for Rheumatoid Arthritis Risk Prediction.

[3] Gao, W.; Tayal, D., Gorinevsky, D.: Probabilistic planning of minigrid with renewables and storage in Western Australia. In: 2019 IEEE Power & Energy Society General Meeting (PESGM) (2019). https://doi.org/10.1109/pesgm40551.2019.8973483

[4] Gao W and Gorinevsky D 2020 Probabilistic modeling for optimization of resource mix with variable generation and storage IEEE Trans. Power Syst. 35 4036–45

[5] Zhang, Yujun, et al. "MamNet: A Novel Hybrid Model for Time-Series Forecasting and Frequency Pattern Analysis in Network Traffic." arXiv preprint arXiv:2507.00304 (2025).

[6] Peng, Q., Planche, B., Gao, Z., Zheng, M., Choudhuri, A., Chen, T., Chen, C. and Wu, Z., 3D Vision-Language Gaussian Splatting. In The Thirteenth International Conference on Learning Representations.

[7] Peng, Qucheng, Ce Zheng, Zhengming Ding, Pu Wang, and Chen Chen. "Exploiting Aggregation and Segregation of Representations for Domain Adaptive Human Pose Estimation." In 2025 IEEE 19th International Conference on Automatic Face and Gesture Recognition (FG), pp. 1-10. IEEE, 2025.

[8] Peng, Qucheng, Chen Bai, Guoxiang Zhang, Bo Xu, Xiaotong Liu, Xiaoyin Zheng, Chen Chen, and Cheng Lu. "NavigScene: Bridging Local Perception and Global Navigation for Beyond-Visual-Range Autonomous Driving." arXiv preprint arXiv:2507.05227 (2025).

[9] Tang, H., Yu, Z., & Liu, H. (2025). Supply Chain Coordination with Dynamic Pricing Advertising and Consumer Welfare An Economic Application. Journal of Industrial Engineering and Applied Science, 3(5), 1–6.

[10] Guo, Y. (2025). The Optimal Trajectory Control Using Deterministic Artifi cial Intelligence for Robotic Manipulator. Industrial Technology Research, 2(3).

[11] Guo, Y., & Tao, D. (2025). Modeling and Simulation Analysis of Robot Environmental Interaction. Artificial Intelligence Technology Research, 2(8).

[12] Zhou, Z. (2025). Research on Software Performance Monitoring and Optimization Strategies in Microservices Architecture. Artificial Intelligence Technology Research, 2(9).

[13] Zhang, T. (2025). Research and Application of Blockchain-Based Medical Data Security Sharing Technology. Artificial Intelligence Technology Research, 2(9).

[14] Yu, Z. (2025). Advanced Applications of Python in Market Trend Analysis Research. MODERN ECONOMICS, 6(1), 115.

[15] Wei, Xiangang, et al. "AI driven intelligent health management systems in telemedicine: An applied research study." Journal of Computer Science and Frontier Technologies 1.2 (2025): 78-86.

[16] Yuan, Yuping, and Haozhong Xue. "Cross-Media Data Fusion and Intelligent Analytics Framework for Comprehensive Information Extraction and Value Mining." (2025).

[17] Chen, J., Zhang, X., Wu, Y., Ghosh, S., Natarajan, P., Chang, S. F., & Allebach, J. (2022). One-stage object referring with gaze estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 5021-5030).

[18] Tan, C., Gao, F., Song, C., Xu, M., Li, Y., & Ma, H. (2024). Highly Reliable CI-JSO based Densely Connected Convolutional Networks Using Transfer Learning for Fault Diagnosis.

[19] Zhuang, R. (2025). Evolutionary Logic and Theoretical Construction of Real Estate Marketing Strategies under Digital Transformation. Economics and Management Innovation, 2(2), 117-124.

[20] Han, X., & Dou, X. (2025). User recommendation method integrating hierarchical graph attention network with multimodal knowledge graph. Frontiers in Neurorobotics, 19, 1587973.

[21] Zhang, Jingbo, et al. "AI-Driven Sales Forecasting in the Gaming Industry: Machine Learning-Based Advertising Market Trend Analysis and Key Feature Mining." (2025).

[22] Zhu, Bingxin. "TaskComm: Task-Oriented Language Agent for Efficient Small Businesses Workflows." (2025).

[23] Zhu, Bingxin. "ReliBridge: Scalable LLM-Based Backbone for Small Business Platform Stability." (2025).

[24] Hu, Xiao. "Learning to Animate: Few-Shot Neural Editors for 3D SMEs." (2025).

[25] Hu, Xiao. "Low-Cost 3D Authoring via Guided Diffusion in GUI-Driven Pipeline." (2025).

[26] Xie, Minhui, and Shujian Chen. "Maestro: Multi-Agent Enhanced System for Task Recognition and Optimization in Manufacturing Lines." Authorea Preprints (2025).

[27] Xie, Minhui, and Shujian Chen. "CoreViz: Context-Aware Reasoning and Visualization Engine for Business Intelligence Dashboards." Authorea Preprints (2025).

[28] Zhu, Bingxin. "RAID: Reliability Automation through Intelligent Detection in Large-Scale Ad Systems." (2025).

[29] Zhang, Yuhan. "SafeServe: Scalable Tooling for Release Safety and Push Testing in Multi-App Monetization Platforms." (2025).

[30] Hu, Xiao. "UnrealAdBlend: Immersive 3D Ad Content Creation via Game Engine Pipelines." (2025).

[31] Li, X., Lin, Y., & Zhang, Y. (2025). A Privacy-Preserving Framework for Advertising Personalization Incorporating Federated Learning and Differential Privacy. arXiv preprint arXiv:2507.12098.

[32] Li, X., Wang, X., & Lin, Y. (2025). Graph Neural Network Enhanced Sequential Recommendation Method for Cross-Platform Ad Campaign. arXiv preprint arXiv:2507.08959.

[33] Xu, Haoran. "CivicMorph: Generative Modeling for Public Space Form Development." (2025).

[34] Tu, Tongwei. "AutoNetTest: A Platform-Aware Framework for Intelligent 5G Network Test Automation and Issue Diagnosis." (2025).

[35] Wang, Hao. "Joint Training of Propensity Model and Prediction Model via Targeted Learning for Recommendation on Data Missing Not at Random." AAAI 2025 Workshop on Artificial Intelligence with Causal Techniques. 2025.