MICRO-FRONTENDS AS A STRATEGY FOR MINIMIZING DOWNTIME DURING TECHNOLOGY MIGRATION

Garifullin R.
bachelor's degree, Saint Petersburg Electrotechnical University «LETI»
Saint Petersburg, Russia
https://doi.org/10.5281/zenodo.15657636

Abstract

This article examines the use of micro-frontend architecture as an effective approach to migrating from legacy web technologies to modern platforms. The methodology of gradual component replacement is explored, enabling continuous system operation and minimizing downtime. Particular attention is given to the practical aspects of implementing micro-frontends, including the division of applications into independent modules, the use of tools for component integration, and maintaining the visual and functional consistency of the interface. The study investigates the impact of micro-frontends on accelerating development, reducing migration risks, and enhancing the resilience of web applications in the face of technological changes.

Keywords: micro-frontends, technology migration, legacy systems, web application modernization, downtime minimization, web application architecture, containerization.

Introduction

The common and fast development of web technologies is a huge challenge to organizations on how to effectively update their systems without causing interruptions to seamless operations. Legacy methods of migration have a lot of downtime, which reduces user experience and leads to inefficiencies in operations. Therefore, this forms a basis on which the adoption of the micro-frontend architecture has become a game-changer.

Micro-frontends can enable progressive replacement of the old components with modern solutions without requiring the rewriting of the whole system from scratch. Because of this architecture, the decomposition of a monolithic frontend into smaller, independently deployable units will guarantee that both legacy and modern technologies will coexist during the migration process. This can reduce system downtime and minimize potential risks resulting from huge migrations. The purpose of this study is to explore how micro-frontend architecture can serve as a strategy for minimizing downtime during technology migration.

Main part. Introduction to micro-frontend architecture and its importance for technology migration

Micro-frontend architecture is a novel approach to web application development in which the user interface is divisible into independent modules. Each module, or micro-frontend, is developed and maintained by a separate team with its own autonomous technology stack. This enables incremental deployment of interface changes without requiring a complete system shutdown. As the technology landscape continues to evolve at a rapid pace and user expectations keep increasing, micro-frontends have become a key strategy to increase the flexibility and agility of web applications.

One of the main problems that developers and organizations confront is migration from old and obsolete technologies to modern ones. As a rule, traditional approaches to migration imply the complete replacement of a system, which is supposed to be connected with considerable time and financial expenses, besides requiring system downtime. This is a very important issue for organizations that provide real-time service, due to the fact that even small cases of downtime result in loss of revenue, loss of brand reputation, and satisfaction among users. Micro-frontend architecture solves all these problems by enabling the integration of legacy and modern components in one system, and thus smoothly transitioning from old technologies to modern ones (table 1).

Table 1 [1, 2]

Comparison of micro-frontend architecture with other approaches

Characteristic	Micro-frontends	Monolithic frontend	SPA (Single Page Application)
Application structure	Divided into independent modules	Unified codebase	Single module with dynamic routing
Development independence	High	Low	Medium
Technology compatibility	Supports multiple frameworks	Limited	Limited
Scalability	High	Limited	High but complex
Component updates	Local (module-specific)	Requires updating the entire application	Requires updating the unified application
Performance optimization	Independent module optimization	System-wide optimization	Dynamic optimization for single app

The application of micro-frontends is especially relevant for large, complex systems where a full migration may take years to complete. Instead of attempting an overall system redesign, micro-frontends facilitate the sequential migration process in which old modules are swapped out with new units without halting the entire app. Additionally, micro-frontend architecture facilitates distributed development where various parts of a project are tackled by independent teams. This parallelised development process simplifies innovation and increases project manageability and flexibility.

Micro-frontend architecture simplifies migration from legacy technologies and provides the potential for easier management of web applications. It helps in reducing the time for updates, minimizing operational risks, and enabling the transition to new technologies even more smoothly.

Methodology of gradual component replacement: approaches and advantages

Migration from other technologies is always accompanied by challenges and risks. Both of these challenges arise due to complexity of function and stability throughout the process of modernization. Unlike the monolithic approach, whereby transformations have a possibility of destabilizing the entire system, micro-frontends allow step-by-step replacement of components and reduce the impact on end users. This approach is based on the seamless integration of new modules into the existing system without requiring a complete system reboot [3].

The process of gradual component replacement begins with identifying the most serious or outdated parts of the web application. These modules typically have a significant impact on performance and user experience. Using micro-frontends, these components can be isolated and replaced with modern solutions while maintaining stable interactions with the rest of the system. Notably, during the migration phase, micro-frontends ensure compatibility between legacy and new technologies. This is achieved through standard interaction interfaces and shared application programming interface (API), which serve as a bridge between outdated and updated modules.

One of the key advantages of the gradual replacement method is the ability to test new components in real-world conditions. Rather than relying solely on testing environments, developers can deploy individual micro-frontends into the existing application and monitor their behaviour in real time. This reduces the likelihood of errors and allows for prompt resolution of identified issues without risking the entire system's operation. This approach provides backward compatibility: if a new component proves incompatible or ineffective, it can be temporarily disabled, and the old version reinstated, something that is often unfeasible with traditional migration methods [4].

Gradual replacement also allows companies to phase out the cost of modernization. Instead of spending a huge amount on total reconstruction at a time, companies can implement changes gradually, which is particularly effective for companies with limited resources. The method also makes project management more efficient with more openness and flexibility in planning. Additionally, the use of micro-frontends enables the simultaneous support of multiple application versions, simplifying the transition process for different user groups. One of the 2024 study shows [5], that this gradual legacy migration to micro-frontends reduces various risks by 72%, increases team productivity by 35%, and achieves cost savings of 45%.

Another advantage is the reduced burden on development teams. Within a micro-frontend architecture, each team can focus exclusively on their specific part of the application, independently of others. This approach can accelerate the replacement process and minimize the risk of conflicts between teams. It is particularly effective for distributed teams or projects involving specialists from different departments or even separate companies.

The incremental component replacement method using micro-frontends is a great strategy for web application modernization. It offers a gentle learning curve to newer technologies, minimizes risks and costs, and allows for the creation of more flexible and fault-tolerant systems. It is a valuable tool for businesses willing to introduce innovations without compromising the stability and quality of their applications.

Practical implementation of micro-frontends for web application modernization

The practical adoption of micro-frontend architecture requires a well-defined strategy and the selection of appropriate tools. Unlike theoretical approaches, the successful implementation of micro-frontends in real-world projects relies on a clear understanding of system requirements, technical constraints, and migration objectives.

One of the initial steps in implementing micro-frontends is determining how to divide the application. This is typically done based on functional domains. In an e-commerce platform, modules for catalog display, shopping cart, and user profile can be identified. Each of these modules can be developed as an independent micro-frontend, allowing teams to build and deploy them autonomously. This approach simplifies the management of complex systems, reduces the likelihood of code conflicts, and accelerates the update process. Figure 1 illustrates the primary approaches to implementing micro-frontend architecture through horizontal and vertical component segmentation.

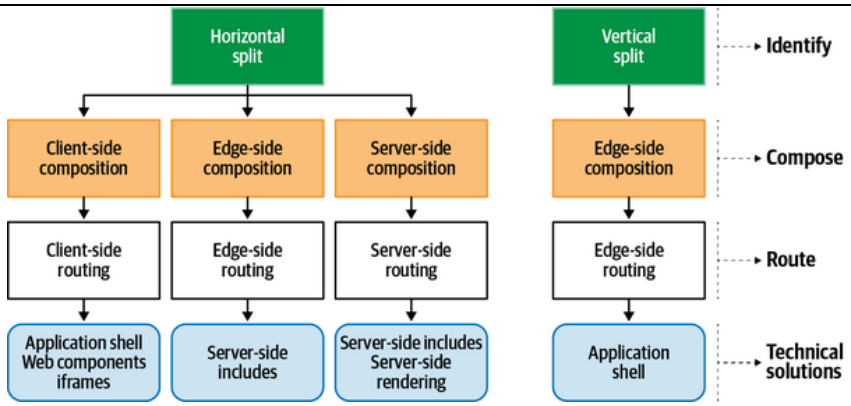

Figure 1. The micro-frontends decisions framework [6]

Horizontal segmentation focuses on layering based on composition methods (client-side, edge-side, server-side), determining where and how modules are integrated. Vertical segmentation, on the other hand, emphasizes the functional aspects of an application, such as user interface, routing, and technical solutions. Each approach boasts strengths and weaknesses, with the freedom to select the most suitable one for specific business requirements. One of these solutions includes client-side routing with Web Components and server-side rendering, and both of these can improve the scalability and performance of a system.

The micro-frontends' technological deployment is typically achieved by leveraging containerization technology such as Webpack Module Federation or Single-SPA. They allow the coexistence of disparate modules within a single application while not compromising on their independence. Webpack Module Federation, for instance, offers dynamic module loading based on their availability, particularly beneficial for massive systems where updating can occur asynchronously [7]. Single-SPA, on the other hand, minimizes state management and routing between micro-frontends and creates the appearance of a unified application for the end user.

Another step of the same magnitude is the consistency of the user interface. Since micro-frontends are being developed by different teams using different technologies, inconsistency in visual style of components and behavior is a possibility. To address this challenge, UI component libraries or design systems are usually implemented that develop common standards for all modules. This way, it is ensured that, in spite of decentralized development, micro-frontends appear and act as if they are components of an integrated application.

Testing micro-frontends requires a specialized approach. In addition to standard unit and integration tests, it is essential to focus on testing interactions between modules. This is a matter of verifying the accuracy of data interchange between micro-frontends, compatibility of their APIs with each other, and interaction when one part is being upgraded. Automated testing has one of the most important roles to fulfill at this juncture, allowing for speedy error identification and correction.

Another important aspect of micro-frontends' real-world use is monitoring them and their state after deployment. Large applications containing numerous isolated components can experience performance issues stemming from a greater number of network requests or uncontrolled resources. Avoiding this, monitoring tools such as Google Lighthouse or specialized Application Performance Management (APM) products should be used. They facilitate tracing the performance of single modules and their contribution to the entire system to be measured.

Major companies actively leverage this architecture to manage their web applications. A sports streaming platform DAZN employed a hybrid approach combining SPA and components managed by a client-side orchestrator called Bootstrap. This method supports web platforms and also smart TVs, set-top boxes, and consoles. The fully client-side orchestrator dynamically loads different SPAs at runtime as users navigate across business domains within the video platform. With this approach, they were able to handle peaks load up to 110000 requests per second [8].

The practical implementation of micro-frontends demands thorough preparation and a comprehensive strategy, but its advantages are clear. It ensures flexibility, scalability, and stability for web applications, allowing organizations to effectively address the challenges associated with modernization. This approach provides extensive opportunities for innovation, making the migration to new technologies both manageable and efficient.

Conclusion

Micro-frontend architecture is a modern and effective way of migrating from legacy technologies to new ones with minimal risk and without system outage. Its ability to allow the retirement of legacy components in place while ensuring interoperability between heterogeneous technologies makes it extremely relevant for large and complex web applications. The practical use of micro-frontends accelerates development, reduces costs, and enhances system reliability through decentralized module control. This strategy enables organizations to react to the fast-changing nature of technology and create more resilient, scalable, and reliable applications that meet the demands of new business and consumers.

References:

10. Peltonen S., Mezzalira L., Taibi D. Motivations, benefits, and issues for adopting micro-frontends: a multivocal literature review // Information and Software Technology. 2021. Vol. 136. P. 106571. DOI: 10.1016/j.infsof.2021.106571

11. Sidorov D. A comparative analysis of component-based architectures in web design for scalable applications // Cold Science. 2024. № 8. P. 24-31.

12. Dudak A. Microservice architecture in frontend development // Norwegian Journal of development of the International Science. 2024. № 145. P. 99-102. DOI: 10.5281/zenodo.14236033

13. Gashi E. The advantages of Micro-Frontend architecture for developing web application // 2024 13th Mediterranean Conference on Embedded Computing (MECO). IEEE. 2024. P. 1-5. DOI: 10.1109/MECO62516.2024.10577836

14. Veeri V. Micro-Frontend Architecture with React: A Comprehensive Guide // International journal of computer engineering and technology (IJCET). 2024. Vol. 15. №. 6. P. 130-153.

15. Building Micro-Frontends. O'Reilly Media, Inc. 2021. 334 p.

16. Bahiense-Junior M. R. G. Micro Frontend Application Shell and Module Federation Architecture Implementation and Comparison // 2024 IEEE International Conference on Data and Software Engineering (ICoDSE). 2024. P. 166-171.

17. Kodall N. Micro frontends: a new paradigm for scalable angular applications // International journal of computer engineering and technology (IJCET). 2024. Vol. 15. №. 5. P. 438-449.