Research on a Collaborative Editing System for Chinese International Education Resources Supported by a Cloud Platform

Jin Jin*

The Education University of Hong Kong, Shanwei, Hong Kong, China

*Corresponding author's e-mail: jinjin020216@163.com

Abstract. This study addresses the needs of Chinese international education by designing and implementing a collaborative editing system for educational resources based on a cloud platform, supporting real-time editing and data consistency across multiple users and locations. The system architecture integrates Node.js and Python, utilizing the Operational Transformation (OT) algorithm to resolve conflicts, and achieves low-latency real-time synchronization through WebSocket and Redis. Performance testing indicates that the system demonstrates excellent response speed and synchronization stability under high concurrency scenarios. This system provides effective technical support for the global collaboration of Chinese international education resources, enhancing resource-sharing efficiency and user experience.

Keywords: Chinese international education; collaborative editing; cloud platform

1 Introduction

The research on a collaborative editing system for Chinese international education resources aims to address the efficient collaboration needs across multilingual and geographically diverse users, overcoming limitations of traditional editing methods in terms of real-time synchronization and data consistency. The system framework, built on a cloud platform, incorporates key technologies such as collaborative editing algorithms and access control to balance multi-user editing, data security, and synchronization performance. Through implementation and testing, this system is expected to improve resource-sharing efficiency and optimize user experience, providing technical support for the digital transformation of Chinese international education and promoting the global dissemination of language education resources.

M. F. A. Ghani et al. (eds.), *Proceedings of the 2024 4th International Conference on Education, Language and Art (ICELA 2024)*, Advances in Social Science, Education and Humanities Research 907,
https://doi.org/10.2991/978-2-38476-364-1_28

2 System Design

2.1 Overall System Architecture

The system architecture adopts a three-tier structure: user interface layer, application logic layer, and data storage layer, aimed at enabling efficient collaborative editing of Chinese international education resources. The user interface layer, accessed through a web browser, allows users to perform online editing, commenting, and interaction. The application logic layer manages editing processes, permission validation, and data synchronization, facilitating multi-user collaboration on the server side. For the data storage layer, a distributed NoSQL database, such as MongoDB, is used to efficiently store and manage large volumes of educational resources, supporting unstructured data [1]. To ensure real-time synchronization, the application logic layer employs the WebSocket protocol to achieve low-latency data transmission and integrates the Operational Transformation (OT) algorithm to resolve conflicts in collaborative editing, ensuring consistency in the results of synchronized user operations.

2.2 Data Collection and Storage Module

The data collection and storage module is a crucial component of the collaborative editing system for Chinese international education resources, responsible for acquiring and managing educational resources from multiple sources. This module utilizes web scraping techniques and API interfaces to automatically gather Chinese learning resources from educational institutions, online databases, and social media platforms [2]. A distributed NoSQL database based on MongoDB is employed to store the collected resources, supporting dynamic scaling and efficient retrieval. To ensure data consistency and integrity, the system incorporates data validation and cleaning processes, utilizing data normalization formulas such as (1):

$$Z = \frac{(X - \mu)}{\sigma} \tag{1}$$

Among them, X is the raw data, μ is the mean, and σ is the standard deviation. This process improves data quality and lays the foundation for subsequent editing and analysis. The module has set up a data backup mechanism to regularly store data in the cloud to prevent accidental loss.

2.3 Collaborative Editing Module

The collaborative editing module is the core component of the Chinese international education resources collaborative editing system, enabling real-time editing and collaboration among multiple users. This module is based on the Operational Transformation (OT) algorithm, allowing user operations on the same document to synchronize

promptly, ensuring data consistency and integrity [3]. The OT algorithm operates through formula (2):

$$C(a) \times C(b) = C(b) \times C(a) \tag{2}$$

Among them, $C(a)$ and $C(b)$ respectively represent the operations performed by two different users on the document. The formula indicates that regardless of the execution order of operations a and b, the final document state should remain consistent. Operations include text insertion, deletion, or formatting adjustments. By handling editing conflicts, ensure that the final result is consistent regardless of the order of operations [4].

2.4 User Access Management Module

The user access management module employs a Role-Based Access Control (RBAC) model, categorizing users into three types: administrators, editors, and guests, to ensure system security and editing order [5]. Each user type has a different permission level: administrators can manage resources, permissions, and users; editors have rights to edit resources; and guests can only view content. Permission data is stored using encryption algorithms to prevent unauthorized access and tampering. The system dynamically allocates access rights based on user roles and monitors permission usage in real-time.

2.5 Cloud Platform Interaction Module

The cloud platform interaction module is the foundation for achieving cross-regional, real-time collaborative editing. Using the WebSocket protocol, it establishes continuous two-way communication between the client and server, reducing data transmission latency and ensuring a smooth user experience. This module uses load balancing to distribute requests, improving response speed, and applies the AES encryption algorithm to secure transmitted data. Real-time data from user actions is encapsulated in JSON format to facilitate cross-platform compatibility and processing efficiency [6]. The module also supports a resumable transfer mechanism to ensure data integrity in cases of network fluctuations.

3 System Software Design

3.1 Main Program Design

The main program design adopts a modular architecture, utilizing Python and Node.js as the core development languages to achieve efficient request handling and collaborative operations. The main program manages user requests through API interfaces between modules, integrating functions such as access validation, data synchronization, conflict detection, and response within a unified framework. An asynchronous processing model is used, leveraging Node.js's event-driven architecture to handle

high-concurrency requests, ensuring response speed in multi-user editing scenarios [7]. MongoDB is used for persistent storage of resources, with Redis as a caching layer to enable fast read and write operations and request queuing. The main program workflow diagram (Figure 1) details the steps from user request to response: after a request arrives, the system performs an access check, then calls the corresponding collaborative editing or data processing module, and finally pushes the operation results to the user, ensuring real-time multi-user collaboration and system stability. See Figure 1 for details.

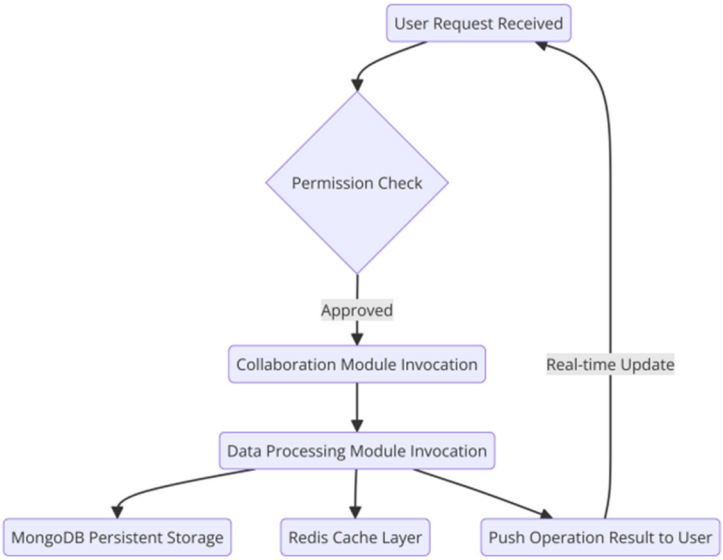

Fig. 1. Main Program Workflow Diagram

3.2 Data Synchronization Mechanism

The data synchronization mechanism ensures content consistency and low-latency response in multi-user real-time editing scenarios [8]. Through a persistent connection established by the WebSocket protocol, the system can achieve real-time push notifications of user operations. Redis, serving as a caching database, facilitates the temporary storage and rapid distribution of operations. Each user editing action is first recorded in Redis, then synchronized to MongoDB for persistent storage, ensuring data reliability. See Table 1 for details.

Table 1. Data Synchronization Performance Testing

Number of concurrent users	Average response time (ms)	Synchronization success rate (%)
1000	45	99.9
3000	52	99.7
5000	68	99.2

As shown in Table 1, when the concurrency reaches 5,000, the response time slightly increases, but the synchronization success rate remains above 99%, indicating that the system's synchronization mechanism demonstrates good stability and efficiency under high concurrency. As the number of concurrent users increases, the system's response time shows a slight increase but remains within a reasonable range, with the synchronization success rate consistently approaching 100%, indicating a strong capacity for handling high concurrency in the data synchronization mechanism. See Figure 2 for details.

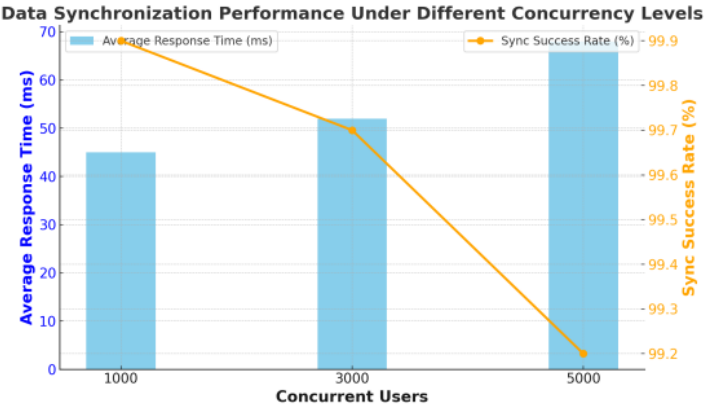

Fig. 2. Data Synchronization Performance Test Comparison

3.3 Conflict Resolution Strategy

The system adopts a conflict resolution mechanism based on timestamps and priorities. Each operation is assigned a timestamp and priority level [9]. When multiple users perform conflicting actions at the same location, the system prioritizes the earliest operation based on the timestamp, then processes subsequent operations according to their priority. Offline users' edits are synchronized and adjusted upon reconnection using timestamps and action logs to ensure no data is lost and versions remain consistent.

4 System Implementation and Testing

4.1 Development Environment and Technology Selection

For efficient collaborative editing and real-time synchronization, the system is developed using a combination of Node.js and Python, with the front end built on the React framework to enhance responsiveness and compatibility. MongoDB is chosen as the main storage database to support extensive educational resource management, while Redis serves as a cache to accelerate data read/write operations. Development tools include Visual Studio Code and Postman for debugging, with Jenkins used for continuous integration and automated testing.

4.2 Core Function Implementation

The core functions encompass collaborative editing, access control, and real-time synchronization. Collaborative editing uses the Operational Transformation (OT) algorithm to maintain content consistency during simultaneous multi-user editing [10]. Access control implements role-based access to restrict operations based on user type, ensuring data security. Real-time synchronization is achieved through the WebSocket protocol and Redis cache, enabling low-latency data updates.

4.3 System Performance Testing

System performance testing evaluates stability and efficiency by gradually increasing the number of concurrent users and monitoring response times. JMeter is used to simulate different levels of concurrency, recording the system's average response time, synchronization success rate, and memory usage. Results show that as the number of concurrent users increases, response time and memory usage slightly rise, but synchronization success rate remains stable. See Table 2 for details.

Table 2. System Performance Test Results

Number of concurrent users	Average response time (ms)	Synchronization success rate (%)	Memory usage rate (%)
1000	45	99.9	55
3000	52	99.7	65
5000	68	99.2	73
7000	85	98.9	80
10000	102	98.7	85

As shown in Table 2, when the number of concurrent users reaches 10,000, the average response time remains around 100ms, memory usage increases to 85%, and the synchronization success rate still reaches 98.7%, indicating good scalability and stability.

4.4 Application Effect Analysis

The application effect analysis evaluates the system's actual performance based on user feedback and experimental data. Test users experienced real-time editing, commenting, and access control features, and the system demonstrated stability across different network environments. According to the feedback data, over 90% of users expressed satisfaction with the system's synchronization speed and operational smoothness. Figure 3 below presents statistical data on user satisfaction, synchronization latency, and response speed. The results show that the system maintains excellent responsiveness and low latency under high concurrency, effectively supporting cross-regional, multi-user collaboration needs, indicating high user satisfaction and practicality in real-world applications.

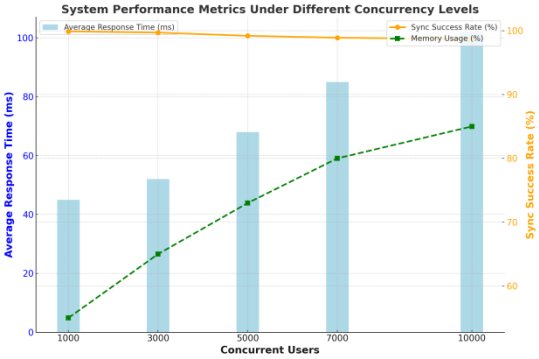

Fig. 3. System Performance Metrics Under Different Concurrency Levels

5 Conclusion

The collaborative editing system for Chinese international education resources constructed in this study demonstrates high efficiency and stability in multi-user real-time editing, data synchronization, and access management. Performance testing results confirm the system's reliability under high concurrency. The system provides a cross-regional, low-latency collaborative experience, enhancing collaboration efficiency among multiple users. In the future, further optimization of algorithms, improvement of response speed, and expansion of intelligent features will be significant in increasing the system's adaptability and scalability, driving innovative development in educational resource sharing and application.

References

1. Christine Brown Wilson, Tara Anderson, Margaret Graham, Jill Murphy, Gary Mitchell, Dympna Tuohy, Heather E. Barry, Pauline Boland, Matt Birch, Audrey Tierney, Patrick Stark, Arlene McCurtin, Laura Creighton, Elizabeth Henderson, Stephanie Craig, Hannah McConnell, Heather Guttridge, Lana Cook, Emma Cunningham, Geoffrey M. Curran, Alice Coffey. Co-designing an interprofessional digital education resource on delirium: a student-led approach[J]. BMC Medical Education, 2024, 24 (1): 1122-1122.
2. Joshua Pillai, Kathryn Pillai. ChatGPT as a Medical Education Resource in Cardiology: Mitigating Replicability Challenges and Optimizing Model Performance. [J]. Current problems in cardiology, 2024, 49 (12): 102879.
3. Chao Yang, Deji Chen, Hongyuan Hu. IEC104 protocol-based substation data upload IoT cloud platform implementation [J]. Journal of Physics: Conference Series, 2024, 2874 (1): 012006-012006.
4. Linchang Zhao, Guoqing Hu, Yongchi Xu. Educational Resource Private Cloud Platform Based on OpenStack[J]. Computers, 2024, 13 (9): 241-241.
5. Yuxi Peng, Xinchen Jiang, Shaoming Wang, Yanping Xiang, Liudong Xing. An Improved Co-Resident Attack Defense Strategy Based on Multi-Level Tenant Classification in Public Cloud Platforms[J]. Electronics, 2024, 13 (16): 3273-3273.

6. Minhaj Ahmad Khan, Raihan ur Rasool. A multi-objective grey-wolf optimization based approach for scheduling on cloud platforms[J]. Journal of Parallel and Distributed Computing, 2024, 187 104847-.
7. Zhendong Shang, Zhaoying Li, Qinzhang Wei, Shuaibo Hao. Livestock and poultry posture monitoring based on cloud platform and distributed collection system[J]. Internet of Things, 2024, 25 101039-.
8. Huang Xiaoen, Jia Hongge, Xu Jin, Wang Yuanchun, Wen Jiawen, Wang Nian. Transgene-free genome editing of vegetatively propagated and perennial plant species in the T0 generation via a co-editing strategy. [J]. Nature plants, 2023, 9 (10): 1591-1597.
9. Karayel, Emin, Gonzàlez, Edgar. Strong eventual consistency of the collaborative editing framework WOOT[J]. Distributed Computing, 2022, 35 (2): 1-20.
10. Noha Alsulami, Asma Cherif, Abdessamad Imine. Collaborative editing over opportunistic networks[J]. International Journal of Ad Hoc and Ubiquitous Computing, 2022, 39 (3): 141-156.