A comprehensive system for the management of e-commerce file and data export information based on domain-driven design

Ke Yang[a], Rihuan Zuo[b], Qiweng Dong[c], Ye Wang[*]

[a]kyang@stu.ecnu.edu.cn, [b]zuorihuan1999@163.com, [c]qwdong@dase.ecnu.edu.cn, [*]ywang@dase.ecnu.edu.cn

School of Data Science and Engineering, East China Normal University, Shanghai 200062, China

Abstract. With the rapid development of Internet technology, e-commerce has thoroughly infiltrated every aspect of commercial activities. It has not only altered traditional business models but also redefined market structures and consumer behaviors. However, existing e-commerce systems exhibit shortcomings in data file management and data export tasks, manifesting as functional gaps and redundant constructions. In response to this challenge, this study designs and implements an e-commerce file task system based on Domain-Driven Design theory and a microservices architecture. This system aims to provide a universal, efficient, and flexible solution for managing the complex and chaotic file and data export tasks inherent in the e-commerce domain. It contributes to enhancing the overall operational efficiency of e-commerce systems and optimizing the end-user experience. Finally, this study validated the effectiveness of the system through well-designed experiments.

Keywords: E-commerce System; Domain-driven design; Microservice

1 Introduction

With the continuous development of computer technology, e-commerce has become an indispensable part of modern business activities. In 2022, 75% of internet users in the European Union had engaged in online purchases or service reservations, marking a 20% increase over the span of a decade [1]. In 2023, China's e-commerce transaction volume reached 46.8273 trillion yuan, with the online retail sales of physical goods amounting to 13.0174 trillion yuan, accounting for 27.6% of the total retail sales of consumer goods [2]. Therefore, constructing an e-commerce system with superior performance and expandability, aimed at providing users with a favorable experience, constitutes a significant goal in the development of e-commerce systems. E-commerce systems frequently face extensive file management demands and intricate, varied data export business scenarios, such as the exportation of bills and product detail lists. How to effectively manage these files and data export tasks is a significant issue in the construction of e-commerce systems. This article, integrating the concepts of Domain-Driven Design (DDD) with the microservices architecture, designs and implements a file task management system for the e-commerce domain. It aims to provide a universal, efficient, and flexible solution for the management of file and data export tasks.

ECIT 2024, March 15-17, Changsha, People's Republic of China

DOI 10.4108/eai.15-3-2024.2346186

2 Related work

Microservices is a software architecture style wherein the software composed of microservices possesses characteristics such as independent deployability and scalability, fulfilling the single responsibility principle of software design [3]. During the implementation of microservices, an event-driven model can be utilized to achieve service decoupling. Producer services and consumer services communicate asynchronously through message pipelines, resulting in reduced dependencies between services, thereby enhancing system performance [4].

Domain-Driven Design (DDD) is a software development methodology that focuses on the core business domain [5]. DDD is commonly employed to guide the construction of microservices architecture systems [6]. Application software developed under the guidance of Domain-Driven Design (DDD) often exhibits functionalities that align with business requirements [7].

3 System design

3.1 Domain model

The system primarily offers two main functionalities: file management and data export task management. The file management component encompasses file upload & download, data access control, and scheduling tasks related to file archiving. On the other hand, data export management is responsible for data exportation, the storage/download of exported files, and the implementation of fallback scheduled tasks to handle failed data export scenarios.

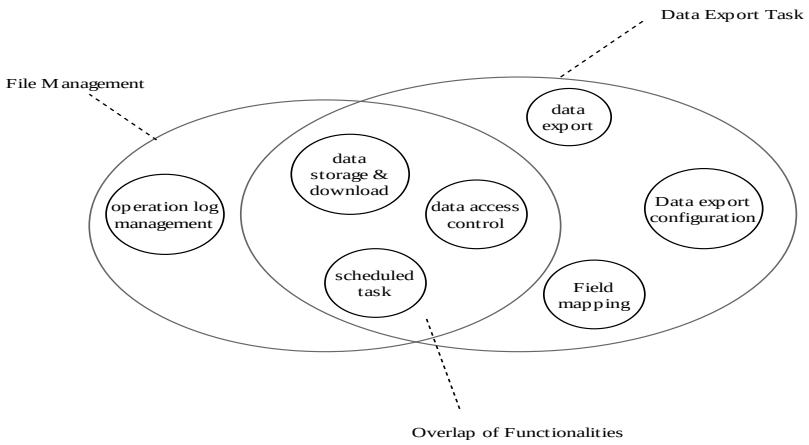

Fig. 1. Overlap of functionlities

Figure 1 illustrates the overlap of functionalities between file management and data export task, as identified through a meticulous process of business workflow analysis and scrutiny. Data export tasks can be regarded as files endowed with a more substantial business context. Consequently, by merging the business functionalities of both domains, the design of the file task domain model is completed. Table 1 presents the details of the domain object consolidation following this analysis.

Table 1. Composition of file task domain model

Aggregation	Domain objects	Domain type
File task aggregation	File task	Aggregation root
Scheduled task aggregation	Scheduled task	Aggregation root
	Scheduled task rule	Value object
Permission aggregation	Permission type	Aggregation root
	Permission rule	Entity
	Permission list	Entity
Metadata aggregation	File task type	Aggregation root
	Export mapping	Entity
	Export rule	Entity
Log aggregation	Operation log	Aggregation root

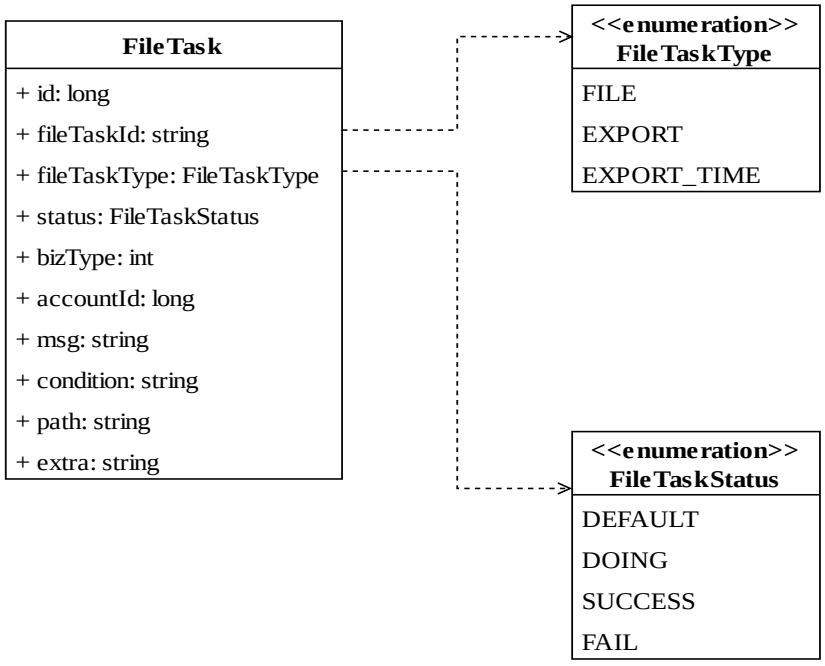

Fig. 2. File task model

Figure 2 illustrates the system's core domain model. FileTask represents the file task, which achieves unified management of file and data export tasks by documenting three types of business fields. Specifically:

(1) **Basic Information**: User identification, business domain, creation time, etc.

(2) **File Attributes**: File storage location, etc.

(3) **Data Export Task Attributes**: Export status, failure reasons, etc.

FileTask distinguishes between file and data export tasks by enumerating FileTaskType and FileTaskStatus.

3.2 System architecture

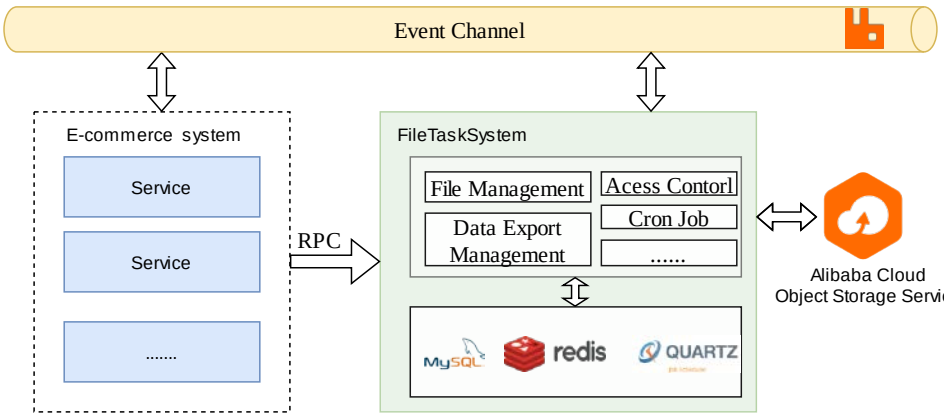

Fig. 3. System architecture

Figure 3 elucidates the system's architectural blueprint, elucidating the symbiotic relationships and interactive modalities among its diverse components. Within this framework, external e-commerce entities engage with the file task management system predominantly through domain events and RPC (Remote Procedure Call) interfaces, facilitating a dynamic and efficient exchange of information. System proffers an array of pivotal functionalities, encompassing file management, data access control, and the orchestration of scheduled tasks. Specifically, the file management module enables robust support for file uploads and downloads. Concurrently, the data access control mechanism plays a critical role in safeguarding sensitive information, underpinning the system's security infrastructure. Moreover, the incorporation of scheduled task management empowers the system to autonomously execute pre-defined operations, thereby amplifying the system's automation quotient. From a technological standpoint, the system leverages MySQL for robust data storage solutions, while Redis serves as the caching middleware, optimizing data retrieval processes and system responsiveness. The Quartz framework undergirds the scheduled task functionality, offering reliable and precise scheduling capabilities. In the realm of inter-component communication, the system's messaging conduit is architected atop RabbitMQ, which ensures efficient and reliable message handling. For file data storage, the system avails itself of Alibaba Cloud's object storage services (OSS), ensuring scalable, secure, and accessible data storage solutions.

4 Functional implementation

4.1 Data access control

The data access control module implements access control management of file task records from three levels: business permission management, permission data query, and authentication components. Among them, business permission management provides the ability to configure permissions for a certain type of business data. Permission data query implementation allows users to only query their authorized access Record the data asked. The authentication component

provides permission filtering for attribute columns and authentication during file downloading for data export scenarios.

(1) **Business permission management**. Administrators can configure data permissions in multiple dimensions according to business types for different business scenarios. When the permission type is set to role (ROLE), the system displays a list of roles for the administrator to choose from. If the permission type is set to specific organizational authorization (UNIT), the system will display the current organizational structure in a tree diagram for the administrator to select. If the file task type is a data export task, the administrator can configure data permissions, setting the visibility of different fields to adhere to the principle of least privilege.

(2) **Permission data query**. To ensure secure data queries, permissions are assigned to file tasks based on their business type. When users perform queries via the API, the type of authorization along with the user's personal information is incorporated as criteria within the query SQL statement. This approach facilitates data queries that are contingent upon permissions. This process is implemented using JPA technology.

(3) **Authentication components**. The authentication component chiefly provides capabilities for permission verification during file data downloads and permission filtering for attribute columns in data export scenarios. As file tasks and metadata attributes utilize a unified set of authorization types, a common algorithmic interface has been developed. This algorithm takes as input parameters the user entity field *user*, the resource type field *type*, and the resource identifier field *id*. The output is a Boolean value *isPassed*. If the resource type is a file task and *isPassed* is true, the file will proceed to download normally. Otherwise, the download process is terminated, and an error message is displayed. If the resource is a metadata column and *isPassed* is true, it indicates that the column is eligible for export, and the data export component will include this attribute column in the exportable fields list. If not, this attribute column will be disregarded.

4.2 File management

System files encompass two distinct attributes: business and physical. The business attributes include the creator of the file, creation time, business domain, and business type, which facilitate file retrieval, browsing, and editing. On the other hand, physical attributes pertain to the file data itself. When designing a file data storage solution, it is imperative to consider the optimization and management of server storage space as well as the future scalability of service instances in terms of data sharing. This approach ensures that the system remains adaptable and can effectively accommodate growth in data volume and evolving access requirements, thereby guaranteeing the stability and reliability of file data storage. The system incorporates Alibaba Cloud's object storage service for file data storage, decoupling business records from file data storage. Compared to on-premises storage solutions (**Figure 4(A)**), this approach (**Figure 4(B)**) not only reduces server load but also alleviates the constraints on service instance expansion caused by file data sharing, thereby enhancing the system's scalability.

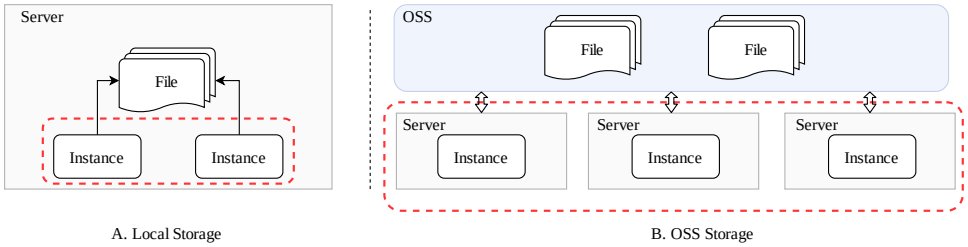

Fig. 4. File storage solution

4.3 Data export

Based on the data export component, the data export process is illustrated in the **Figure 5.** The data export execution process can be triggered by users via API calls or by system calls. Specifically:

1. The entry parameter for the data export task executor is the file task identifier task_id. The export executor initially invokes the *findFileTaskById* interface of the FileTaskDomainService, which is a file task domain service, to query the file task record.

2. After retrieving the corresponding file task record, the user identifier on the record is used to fetch user information through an RPC service call to the user query interface.

3. Based on the user information and file task information obtained from steps 1 and 2, the file export task context is created using the context factory class, ExportContextFactory.

4. The MetadataDomainService, a metadata domain service, is invoked using the *bizType* field to obtain metadata configurations through the *findMetadataByBizType* interface. Concurrently, the *findBizPermissionRuleByBizType* interface is called to acquire the business permission rules corresponding to the business type of the export task.

5. If the *columnPermission* flag obtained in step 4 is set to true, indicating that column permission filtering is enabled, the process proceeds to step 6; otherwise, it moves to step 7.

6. Invoke the *columnFilterByPermission* method provided by the PermissionProcessor authentication component, inputting user information and metadata information, to obtain the exportable attribute columns that have been filtered through permission checks.

7. The data export component uses the *dataHandler* field from the metadata configuration to invoke the dataHandlerFactory, which generates a predefined data processor. The primary function of the data processor is to query the data to be exported and apply data processing based on the metadata configuration rules. Finally, the *exportToInputStream* method is called to complete the generation of the data export file and upload it to the object server. The system proposes a parallelization improvement to the original sequential data export scheme by introducing the Java BlockingQueue. This approach replaces the serial process of data querying, processing, and exporting within the process with parallel thread read/write operations. The multi-threaded read/write scheme enhances CPU utilization per unit time, optimizing the file export duration.

8. After completing the data export and file upload, the corresponding file task's status field, Status, is updated to indicate success.

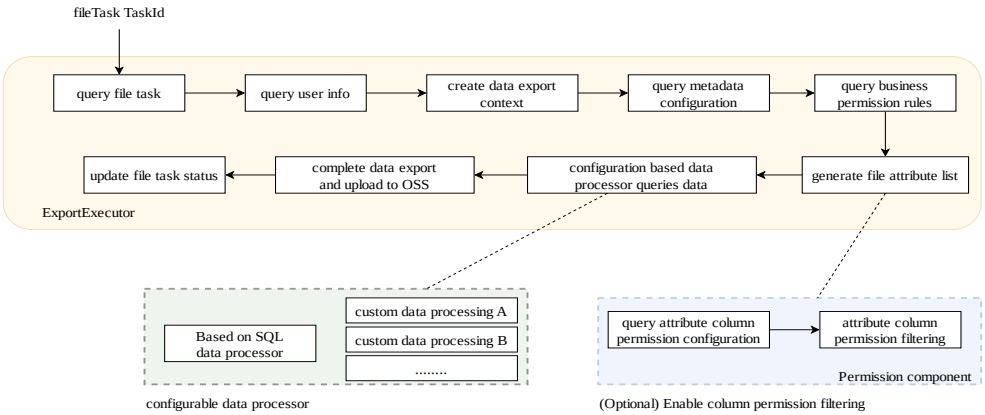

Fig. 5. Data export process

4.4 Log management

Operation records are generated asynchronously based on domain events, ensuring that the asynchronous recording does not affect the main business process and achieving decoupling between log recording actions and business operations. Administrators can view system operation logs on the log management page. The operation record module systematically records and stores activity information related to file tasks, enabling traceable management of data operations. This functionality allows administrators to scrutinize the utilization of file tasks through log analysis, thereby enhancing the supervision and protection of sensitive information.

4.5 Cron job

During the operation of a system, there are often two types of unforeseen circumstances: (i) Due to system anomalies, data query timeouts result in data export tasks that are continuously in an exporting state or fail to export. (ii) Temporary data needs to be regularly cleaned to avoid occupying space or the risk of slow queries. The system addresses the aforementioned issues through two types of scheduled tasks:

(1) **Compensation tasks**. It scans for data export tasks with a status of unsuccessful under a certain business type and triggers the data export component to re-execute the task.

(2) **Archival Task**. It scans for data that meets the archival criteria under a certain business type, synchronizes it to a cold table, and then executes a batch delete command.

5 System test

To ensure the reliability of the system test results, the configuration of the test environment is consistent with the production environment, as shown in Table 2.

Table 2. Interface response time testing

Type	Configuration
Operating System	CentOS 7.4.1708 64bit
Processor	Intel(R) Xeon(R) Gold 6142 CPU @ 2.60GHz, 8core
Hard Drive	256GB
Memory	16GB
Network	Operation log
Testing Tool	VisualVM, Apache JMeter

The goal of performance testing is to verify the system's responsiveness, stability, and reliability in a specific environment. This section focuses on testing the interface response time, throughput of the file task system, and the performance during the file export scenario.

5.1 Interface response time testing

Table 3 shows the response time test results for some key interfaces of the system using Apache JMeter. The test was configured with 600 threads to simulate a scenario where 600 concurrent users access the interface under a daily use case. The actual response times in the table use the 95th percentile from the aggregate report as the metric, indicating that during the test, 95% of the request response times were below this value, thereby representing the actual experience of the vast majority of users. The test results indicate that the system interface response performance meets the expected criteria.

Table 3. Interface response time testing

Interface Name	Description	Expected Response Time	Line 95%	Conclusion
getFileTaskById	File Task Query Interface	400ms	160ms	✓
getDownloadUrl	File Download Link Retrieval Interface	400ms	243ms	✓
searchLog	Operation Log Query Interface	800ms	743ms	✓
fileTaskList	File Task List Retrieval Interface	800ms	567ms	✓

5.2 Throught testing

Table 4. Interface throughput testing

Interface Name	System	Original Solution
Data Export Task Creation	103/s	45/min

Throughput is a critical metric for measuring a system's capacity to process requests within a unit of time. Table 4 presents the throughput test case for the data export task creation process in the file task system, comparing it with the original solution. The data export task chosen for creation involves a task with 350 rows of data. Since the file task system processes data export requests asynchronously based on domain events, the throughput is represented by the acknowledgment rate (ack rate, or consumption efficiency) in the RabbitMQ console. The

experimental results demonstrate that, compared to the original scheme, the file task system possesses superior request processing capabilities, making it more capable of handling high-concurrency scenarios.

5.3 Performance testing of data export scenarios

The implementation of the file task center has unified the file management capabilities of the information system, centralizing it on the middle platform while also decoupling the data export function of the business system. This optimization prevents the scenario where a large amount of data resides in the business system's memory during data export tasks, thereby reducing the potential impact on the stability of the business system. This experiment aims to evaluate the performance advantages of the business middle platform solution by comparing the stability of the business system and the data export duration with the original system solution when processing data export tasks with different data volumes (100, 500, 1000, 2000, 5000, 10000 rows of data).

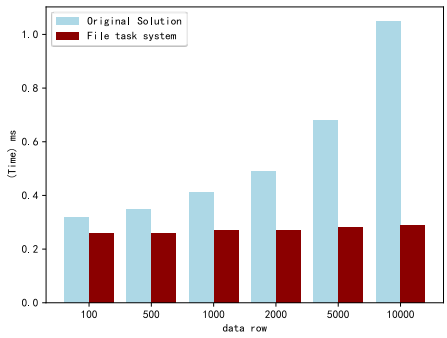

Fig. 6. E-commerce system memory usage

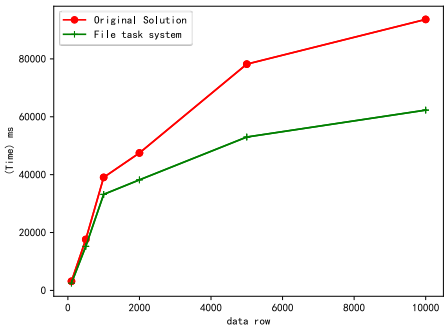

Fig. 7. Data export duration

Figure 6 illustrates the impact of the solution and the file task system on the memory usage of the e-commerce system under different data export volumes. The test results indicate that after integrating the file task system, the memory usage of the business system is significantly

optimized. Figure 7 shows the comparison between the original solution and the file task system in terms of data export duration. The test results demonstrate that the business middle platform solution excels in reducing the time required for data export, thereby enhancing user experience.

6 Conclusion

This paper designs and implements a general file task information system for the e-commerce domain, aimed at unified management of file and data export tasks generated by different e-commerce business systems. During the system design phase, the paper identifies system domain objects and performs business modeling based on DDD (Domain-Driven Design) theory, completing the design of the system domain model and the planning of the system architecture. Subsequently, the paper provides a detailed description of the system implementation. Through the data access control module, administrators can easily grant file task authorizations for different business types and achieve data permission control to prevent the leakage of confidential information. The file management module, leveraging object storage services, enables shared storage of file data, ensuring system scalability. The data export module offers a flexible configuration and high-performance data export interface. The log management module facilitates asynchronous recording of file task operations, allowing for traceable management of data operations. Finally, the implementation of the Cron Job module automates the operation and maintenance of the system by regularly processing unexpected file tasks. Finally, this paper devises a comprehensive testing plan to assess the interface response performance, throughput, and performance in data export scenarios of the file task system. The experimental results confirm the excellent response performance of the file task center and demonstrate a significant performance improvement compared to the original solution.

Acknowledgments. These authors are contributed equally to this study. This work is partly supported by the National Natural Science Foundation of China (62002119,61502236, 61672234, 61877018, 61977025, U1811264), the National Key Research and Development Program of China(2016YFB1000905), Shanghai Agriculture Applied Technology Development Program, China(T20170303).

References

[1] Sulova S. A conceptual framework for the technological advancement of e-commerce application s[J]. Businesses, 2023, 3(1): 220-230.

[2] National Bureau of Statistics. Statistical Bulletin on National Economic and Social Development of the People's Republic of China in 2023 [EB/OL]. 2023. https://www.stats.gov.cn/sj/zxfb/202402/t20240228_1947915.html.

[3] Zimmermann O .Microservices tenets Agile approach to service development and deployment[J]. Computer science, 2017, 32(3-4):301-310.

[4] Singjai A, Zdun U, Zimmermann O. Practitioner views on the interrelation of microservice apis and domain-driven design: A grey literature study based on grounded theory[C]//2021 IEEE 18th International Conference on Software Architecture (ICSA). 2021: 25–35

[5] Evans.Domain-Driven Design: Tacking Complexity In the Heart of Software[M]. 2004.

[6] Chuangxin Ou, Di Deng. Middle platform architecture and implementation based on DDD and microservices[M]. China Machine Press, 2020.
[7] Dr. Faisal Shahzad, Prof. Ricardo Martinez. Domain-driven design: Aligning software with business needs for improved agility[J]. INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY, 2023, 7(4): 15–21.