Database Bundle System with High Availability: Employing Message-Oriented Intermediary Software

J.S.Jaslin,
Assistant Professor, Department of Computer Science and Engineering, J.J. College of Engineering and Technology, Trichy, Tamilnadu
S.Harthy Ruby Priya,
Assistant Professor, Department of Computer Science and Engineering, J.J. College of Engineering and Technology, Trichy, Tamilnadu
Dr.M.P.Revathi,
Professor, Department of Computer Science and Engineering, J.J. College of Engineering and Technology, Trichy, Tamilnadu
T.Vency Stephisia,
Assistant Professor, Department of Computer Science and Engineering, J.J. College of Engineering and Technology, Trichy, Tamilnadu

Abstract

This research presents a high-availability database bundle system that employs message-oriented intermediary software. The system consists of a database service layer comprising multiple database servers and a middle layer situated between the application layer and the database service layer. The middle layer incorporates application proxies, database intermediary software, and a message bus. The application proxies enable connection redirection in the event of node failures, ensuring uninterrupted service and enhanced reliability. The database intermediary software is responsible for intercepting, analyzing, processing, and retransmitting database request messages. The message bus facilitates global sorting of read and write operations performed by the data intermediary software, ensuring data consistency and delivering the sorting results to the database servers. The proposed system offers a practical solution for achieving a high-performance database with excellent availability. It exhibits advantages such as high availability, scalability, and easy construction and management.

Keywords: high-availability, database bundle, message-oriented intermediary software, application proxies, database intermediary software, message bus, data consistency, reliability, scalability, management

DOI Number: 10.48047/nq.2020.18.8.nq20223 **NeuroQuantology 2020;18(8):179-184**

Introduction

In today's digital era, organizations heavily rely on high-performance and highly available database systems to support critical applications such as e-commerce, train ticket ordering services, and e-banking. These applications often experience a surge in user traffic, with peak inquiry rates exceeding 100,000 times within short periods. Meeting the performance requirements of such demanding workloads is a significant challenge for traditional single-server systems in terms of processing power and network bandwidth.[1]

The inefficiency or failure of servers during peak periods can lead to substantial losses for service providers. As a result, there is a growing interest in the research and

development of database bundle technologies within both academia and industry. Database bundleing offers a promising solution by distributing the workload across multiple servers to enhance performance, scalability, and fault tolerance.Existing database bundle systems can be broadly classified into shared storage organization and unshared structure.[2] Shared storage organizations demonstrate faster access to data due to specialized memory devices. However, these systems tend to be costly and may limit scalability. On the other hand, unshared bundle topologies, which employ storage sub-bundles, offer high extensibility but are often limited to specific database engines and involve complex system architectures.

Due to the constant evolution of businesses, institutions, and technologies, it is crucial for software systems to be adaptable to these changes. Whether it is due to mergers, the introduction of new services, or the expansion of existing ones, organizations cannot afford to rebuild their information systems from scratch. Instead, they need to integrate new components or scale existing ones efficiently. The most convenient approach to integrating diverse components is not by transforming them into homogeneous elements, but by establishing a communication layer that allows them to interact despite their differences. This intermediary layer, known as middleware, enables independent software components (such as applications, enterprise Java beans, servlets, and others) running on different networked platforms to seamlessly communicate with one another. When this interaction becomes possible, the network can truly function as a unified computing system.**Figure 1** illustrates the conceptual placement of middleware, positioned between the application layer and the platform layer.

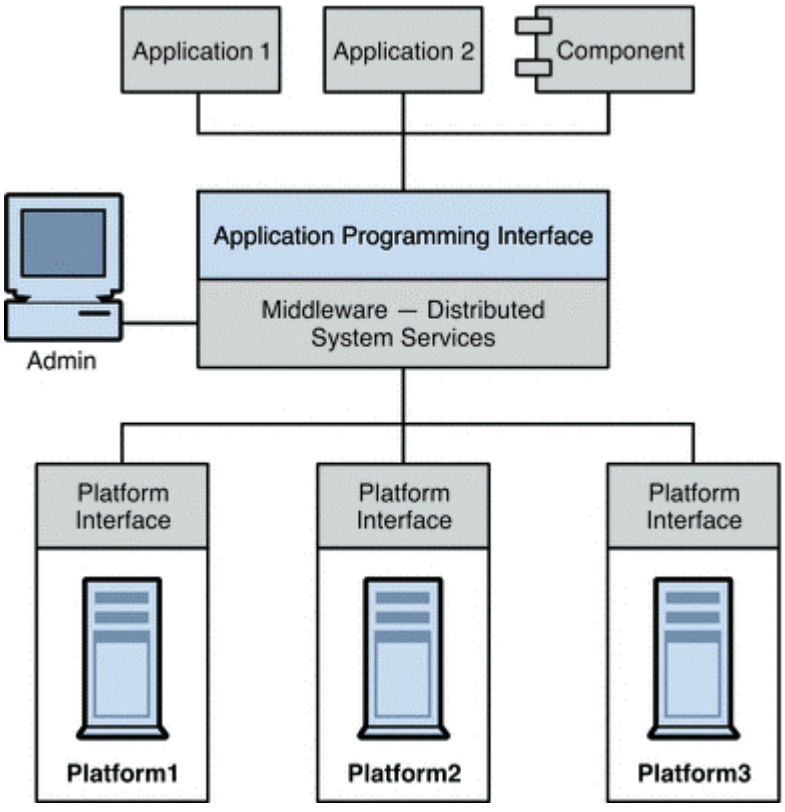

Figure 1. Message-Oriented Middleware (MOM)

Furthermore, fault tolerance in network connections, particularly at the TCP level, remains a challenge in most existing database bundle systems. While certain proprietary technologies like Oracle RAC address this issue, a comprehensive solution with high availability is still sought after.In this context, this research focuses on developing a high-

availability database bundle system based on message-oriented intermediary software. The proposed system comprises a database service layer with multiple database servers and a middle layer situated between the application layer and the database service layer.[3] The middle layer consists of application proxies, database intermediary software, and a message bus, each serving specific functions to ensure uninterrupted service, improve reliability, and maintain data consistency.

Middleware can be categorized into three distinct groups. These models enable one software component to influence the behaviour of another component across a network. However, they differ in the sense that RPC- and ORB-based middleware create tightly-coupled component systems, while MOM-based systems allow for a more loosely coupled integration of components. In an RPC- or ORB-based system, the calling procedure must wait for the called procedure to return before proceeding with any other actions. Within these synchronous messaging models, the middleware functions as a super-linker of sorts, identifying the location of the called procedure on the network and utilizing network services to transmit function or method parameters to the procedure, as well as returning the results.MOM-driven systems enable communication by facilitating the asynchronous exchange of messages, as depicted in **Figure 2**.

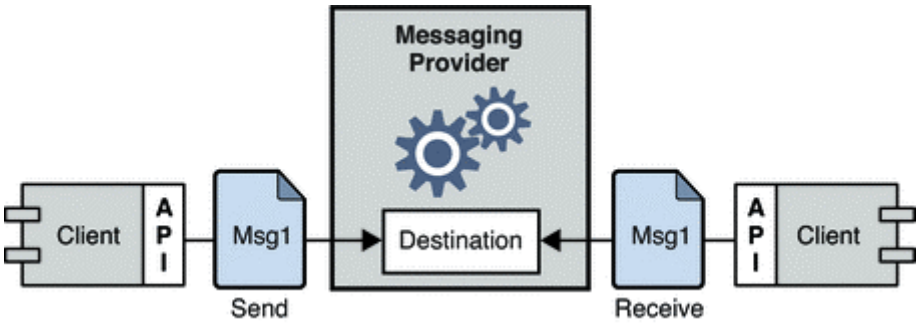

Figure 2. MOM-Based System

The primary objective of this research is to provide a feasible approach for organizations to obtain a high-performance database system with high availability. The system's advantages include not only high availability but also excellent scalability, easy construction, and management.By addressing the limitations of existing database bundling approaches, this research aims to contribute to the advancement of reliable and efficient database systems capable of handling demanding workloads.The following sections will delve into the details of the proposed high-availability database bundle system, including its architecture, functionalities, and the benefits it offers.[4] This research aims to provide valuable insights and practical solutions for organizations seeking to enhance their database infrastructure and ensure reliable and efficient operations in the face of increasing data demands and user traffic.

Related Work

In today's popular applications such as e-commerce, train ticket ordering services, and e-banking, there are instances where the number of users surges within a short period of time. The peak number of inquiries can exceed 100,000 times. However, the processing power of a single server, whether in terms of CPU performance or network bandwidth, is unable to meet such high-performance requirements. Consequently, server inefficiency can result in significant losses for service providers. This situation has prompted academia and industry to focus on studying and developing database bundle technologies.[1]

Currently, database bundle products are primarily categorized into shared storage organization and unshared structure. Shared storage organization bundles, which employ special-purpose memory devices, have higher expectations and faster access speeds for

database server data. However, the use of these specialized memory devices increases system construction costs and limits the scalability of the bundle.On the other hand, unshared bundle topologies, which employ storage sub-bundles similar to parallel file systems, offer excellent scalability.[3] However, most of these database bundles are based on specific database engines like MySQLBundle, which can only employ certain storage engines such as NDB Bundle and cannot incorporate widely used engines like MyISAM or InnoDB. Additionally, these systems have complex architectures, resulting in high development and maintenance costs.Furthermore, existing database bundles generally perform poorly in terms of network fault tolerance, particularly at the TCP connection level.[6] While Oracle RAC has some TCP fault tolerance capabilities, this technology is proprietary. In contrast, the present invention proposes a high-availability database bundle system based on message-oriented intermediary software, which effectively addresses all the issues.

By leveraging message-oriented intermediary software, this system overcomes the limitations of previous solutions. It provides excellent fault tolerance, scalability, and compatibility with a wide range of database servers and storage engines. The system's architecture simplifies development and maintenance, and its use of message-oriented intermediary software ensures reliable and efficient communication between components. Overall, this high-availability database bundle system offers an effective and comprehensive solution for organizations seeking robust and reliable database systems.

Research Objective

The objective of this research is to develop a high-availability database bundle system based on message-oriented intermediary software. The system aims to provide uninterrupted service, improved reliability, and data consistency by implementing connection redirection, message interception and analysis, and global sorting of database operations. The research focuses on addressing the challenges of achieving high availability and scalability in database systems while ensuring data integrity. The objective is to propose a feasible approach that combines the benefits of message-oriented intermediary software and database bundleing to create a high-performance and highly available database system.

Database Bundle System with High Availability: Employing Message-Oriented Intermediary Software

The high availability database bundle system, which is based on message-oriented intermediary software, includes a database service layer consisting of M platform database servers (4). It is characterized by the presence of a middle layer positioned between the application layer and the database service layer (4). This middle layer comprises N application proxies (2), M database intermediary software (3), and a messaging bus, where 1≤N≤16 and 1≤M≤8.The application proxies (2) redirect connections when service nodes experience failures, ensuring uninterrupted service and improved reliability. The database intermediary software (3) intercepts, parses, processes, and forwards the database request messages. The messaging bus facilitates communication between the database intermediary software (3) and the database servers (4) and is responsible for globally ordering the read and write operations sent by the data intermediary software, ensuring data consistency. The ordering results are then sent to the database servers (4).The database intermediary software (3) consists of various modules, including the message sink module (3A), MySQL message resolution module (3B), SQL statement parsing module (3C), request execution and result handling module (3D), message processing module (3E), fault detection module (3F), data recovery module (3G), log pattern (3H), group communication module (3I), and concurrent control module (3J).

The message sink module (3A) listens for port information and receives requests from the upper layer of the database. It initializes and interrupts connections from the established MySQL connection pool and handles SQL

commands for database services, forwarding them to the MySQL message resolution module (3B).The MySQL message resolution module (3B) resolves the MySQL messages received from the message sink module and identifies their types. It decompresses and analyzes the messages to obtain SQL commands, which are then sent to the SQL statement parsing module (3C) for further processing.The SQL statement parsing module (3C) retrieves MySQL message read and write orders from the MySQL message resolution module (3B) and parses their types. If it's a non-write order, the request is directly forwarded to the request execution and result handling module (3D). If it's a write order, the analysis result is returned to the message processing module (3E).

The request execution and result handling module (3D) directly send local node read messages to the database server (4) for execution. Global write orders are sent to the concurrent control module (3J), and the comparison of the execution results is also returned to the application proxy (2).The message processing module (3E) is responsible for processing and forwarding messages within the system. It places the processed information on the messaging bus through the group communication module (3I) for transmission to other backends. It also receives messages from other backends via the group communication module (3I). For write orders that include global ordering, it parses the MySQL message request and executes it using the request execution and result handling module (3D). The log pattern (3H) is used to record updates, including the completion of daily records before sending and after receiving results. In case of failure messages, the fault detection module (3F) is called upon for analysis and processing.The fault detection module (3F) employs the information obtained from the message processing module (3E) to identify changes in the system's member set. It analyzes and processes system faults, and in the event of a new node addition or a malfunctioning node, the data recovery module (3G) performs data synchronization.The data recovery module (3G) employs the log pattern (3H) to analyze the updates recorded in the daily record. It identifies data differences between newly added nodes, recovered nodes, and normal nodes, and synchronizes the data to achieve database consistency across all nodes.The log pattern (3H) serves the purpose of recording updates and defines the format for incremental recovery scripts. It provides an access interface for both the message processing module (3E) and the data recovery module (3G) to retrieve and employ the recorded daily records.The group communication module (3I) acts as an interface between the message processing module and the messaging bus. It facilitates communication and the exchange of messages within the system. The concurrent control module (3J) employs a distributed lock table to ensure the execution of requests and the overall sequence of transactions are consistent and coordinated.

In summary, the high availability database bundle system based on message-oriented intermediary software consists of a layered architecture with application proxies, database intermediary software, and a messaging bus. It ensures uninterrupted service by redirecting connections during node failures, intercepts and processes database request messages, maintains data consistency through global ordering, and provides scalability, fault tolerance, and easy management. The system's extensive modules, such as fault detection, data recovery, and concurrent control, contribute to its high availability, scalability, and reliability, making it a feasible and advantageous solution for building high-performance databases.

Conclusion

In conclusion, this research presents a high-availability database bundle system based on message-oriented intermediary software. The system incorporates application proxies, database intermediary software, and a message bus to ensure uninterrupted service, improve reliability, and maintain data consistency. The proposed system offers advantages such as high availability, good scalability, and easy construction and

management. By combining the strengths of message-oriented intermediary software and database bundleing, the research provides a feasible approach for achieving a high-performance database with excellent availability. The system's effectiveness and potential benefits make it a valuable solution for organizations that require robust and reliable database systems.

Reference

1. Lilis, G., &Kayal, M. (2018). A secure and distributed message oriented middleware for smart building applications. Automation in Construction, 86, 163-175. https://doi.org/10.1016/j.autcon.2017.10.030
2. Plaza, A. M., Díaz, J., & Pérez, J. (2018). Software architectures for health care cyber-physical systems: A systematic literature review. Journal of Software: Evolution and Process, 30(7), e1930. https://doi.org/10.1002/smr.1930
3. J. Rodríguez-Molina and D. M. Kammen, "Middleware Architectures for the Smart Grid: A Survey on the State-of-the-Art, Taxonomy and Main Open Issues," in IEEE Communications Surveys & Tutorials, vol. 20, no. 4, pp. 2992-3033, Fourthquarter 2018, doi: 10.1109/COMST.2018.2846284.
4. J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang and W. Zhao, "A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications," in IEEE Internet of Things Journal, vol. 4, no. 5, pp. 1125-1142, Oct. 2017, doi: 10.1109/JIOT.2017.2683200.
5. S. Kugele, D. Hettler and J. Peter, "Data-Centric Communication and Containerization for Future Automotive Software Architectures," 2018 IEEE International Conference on Software Architecture (ICSA), Seattle, WA, USA, 2018, pp. 65-6509, doi: 10.1109/ICSA.2018.00016.
6. P. Kathiravelu and L. Veiga, "SD-CPS: Taming the challenges of Cyber-Physical Systems with a Software-Defined approach," 2017 Fourth International Conference on Software Defined Systems (SDS), Valencia, Spain, 2017, pp. 6-13, doi: 10.1109/SDS.2017.7939133.
7. A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," in IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347-2376, Fourthquarter 2015, doi: 10.1109/COMST.2015.2444095.
8. Boyer, J., Mili, H. (2011). Issues in Designing Business Rule Applications. In: Agile Business Rule Development. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-19041-4_7