Design of Event Management System for Space Big Data Secure Transmission Message Middleware Based on G/S mode

Yuanhong Zhang[1,a], Min Zhang[2,b,*]

[a]873158916@qq.com,[*]Corresponding author:[b]362631406@qq.com

[1]School of Computer Engineering, Chengdu Technological University, Chengdu 611730, China
[2]School of Network and Communication Engineering, Chengdu Technological University, Chengdu 611730, China

Abstract. With the continuous expansion of the scope of spatial information applications, the G/S model has achieved a new leap in network access to spatial information from the current C/S and B/S structures. As an important part of communication between G and S ends in G/S mode, message middleware plays a particularly important role in ensuring the security of spatial data transmission. This paper proposes a message middleware system in G/S mode, which ensures the security of spatial data transmission and effectively ensures the network efficiency of the entire G/S mode spatial data security middleware system. One important component of Messaging-Oriented Middleware is the Event Management Subsystem (EMS). The thesis ends with the discussion on the designing and implementing of the Event Management Subsystem of Messaging-Oriented Middleware.

Keywords: Message-Oriented Middleware; Event; Spatial Data; Global State

1. INTRODUCTION

In 2004, an article titled "Mapping Opportunities" [3]in the American journal Nature identified Geotechnology, biotechnology, and nanotechnology as the three cutting-edge sciences of the 21st century. Due to the advent of the Neogeography era, more people are learning to use spatial and geographical thinking to serve themselves. With the application of Global Positioning System (GNSS), various professional geographic information platforms, digital cities, and digital earth construction, the demand for spatial data is becoming increasingly strong. The G/S mode is a spatial data exchange standard based on the new hypergraphic (or hypergeographic) markup language HGML (Hypergraphic Markup Language or Hyper Geographic Markup Language), where G is the spatial information browser and S is the spatial information server. Middleware is the core of future cloud services [2], located between application software and operating systems, at the lower level of application software, above spatial databases, networks, and operating systems. It is mainly used to help users efficiently and flexibly integrate and develop complex application software[5]. Due to the extremely complex nature of spatial data, coupled with the large number and dynamic changes of users and resources in the system environment, new challenges and opportunities for spatial data security have emerged. Integrating middleware technology with the new G/S model to achieve

PMBDA 2023, December 15-17, Nanjing, People's Republic of China

DOI 10.4108/eai.15-12-2023.2345421

multi-level security such as system connections, application programs, and message encryption and decryption[7], and researching the security mechanism of spatial data transmission in its architecture system, providing theoretical basis and application models for exploring and establishing the availability of a service-oriented popular spatial information service application platform in the new geographic information era[6].

The main task of Message-Oriented Middleware in event management is to monitor the status of the entire system, and when the state change notification system or application. Message-Oriented Middleware event management subsystem to complete the following two main functions:

Event generation and notification.

System status monitoring and the establishment of a global state.

The Event Management Subsystem of Messaging-Oriented Middleware implementation usually with reference to (but not strictly follow) the SUN's distributed event specification[1], the following is a detailed discussion to the Event Management Subsystem of Messaging-Oriented Middleware to achieve.

2. MESSAGE-ORIENTED MIDDLEWARE EMS DESIGN AND IMPLEMENTATION

Because the Message-Oriented Middleware can not follow uniform standards for the design and implementation of Message-Oriented Middleware with different middleware providers will be different. Similarly, in the realization of Message-Oriented Middleware event management subsystem not to follow the course of the design and implementation specifications; But the design and implementation of Message-Oriented Middleware event management subsystem in the light of the distributed event specification, Java Message Service specification subscription publishing model; In addition, Message-Oriented Middleware event management subsystem also refer GSRA algorithm for global state of the records, systems and applications that need to be logged in to use this global state.

2.1 Message-Oriented Middleware EMS Workflow

Message-Oriented Middleware EMS workflow shown in Figure 1:

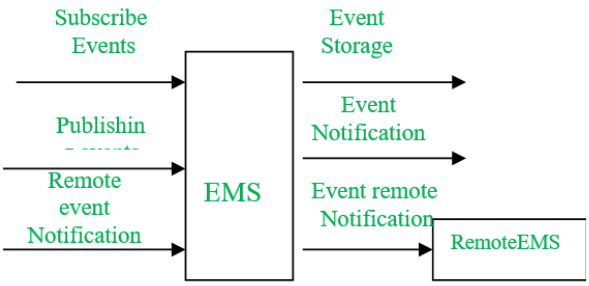

Figure 1. Workflow messaging middleware EMS

Seen from the figure, EMS to be responsible for the completion of the function:

- Receive Subscribe to local event.
- Receive local event producers publish events.
- Receive remote EMS notification.
- Storage of events (such as dead-letter event, etc.).
- The local application event notification.
- Forward event notifications to the remote application (via remote EMS).

EMS this structure shows that it implements the so-called distributed event specification in third-party object function[8].EMS through the development of the same event-related strategies to achieve the filtering of the event; In addition, it uses the concept of the event queue, it has a storage function of the incident (but such storage is limited), so the EMS to achieve a distributed event specification storage - transmitting agent, the event filter, Event-mail and other third-party objects need to implement the function.

However, despite the Message-Oriented Middleware EMS reference to distributed event specification, it is not fully to achieve in accordance with specifications distributed events, and should not be distributed event specification from the point of view to discuss Message-Oriented Middleware EMS functionality.

2.2 Message-Oriented Middleware EMS Module Structure and Realization

Here is the Message-Oriented Middleware EMS framework for the entire module structure, and discuss its concrete realization. Message-Oriented Middleware EMS have achieved on the Win NT/9x/2000 and UNIX; Because different operating systems, Message-Oriented Middleware EMS to achieve different operating systems use different methods and techniques. These technical details do not do a detailed discussion.

Message-Oriented Middleware EMS module structure as shown in Figure 2:

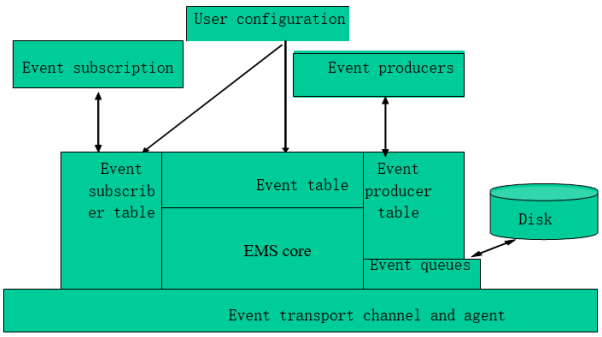

Figure 2. Schematic diagram of messaging middleware EMS module

Message-Oriented Middleware EMS event producers are divided into two categories:

Core Program: Message-Oriented Middleware system itself

Applications: other programs built on top of Message-Oriented Middleware

In order to distinguish between the different time producer, Message-Oriented Middleware EMS achieve description of the event producer by producer object.

Similarly, Message-Oriented Middleware, a structure used to describe the event subscribers; In order to achieve subscribe / publish model, in addition to the structure that contains the subscriber's own information, but also includes how to notify the subscriber and other information.

System maintains a subscriber table and a producer table, subscriber table is the role of record of the event to subscribe subscriber information; EMS that there are two Subscribers: local subscribers and remote subscribers, in terms of local subscribers, EMS allows the subscription are: notification mode (push) and access mode (pull).Notice mode of subscribers, it will provide its subscription event notification means (such as the WINDOWS, this may be a message ID and window handle, to the UNIX case may be a pipe or a message queue identification).

To understand these objects, you must first understand the event subscription process and the publishing process of applications program based on Message-Oriented Middleware EMS. The following Figure 3 shows the outline of the subscription process:

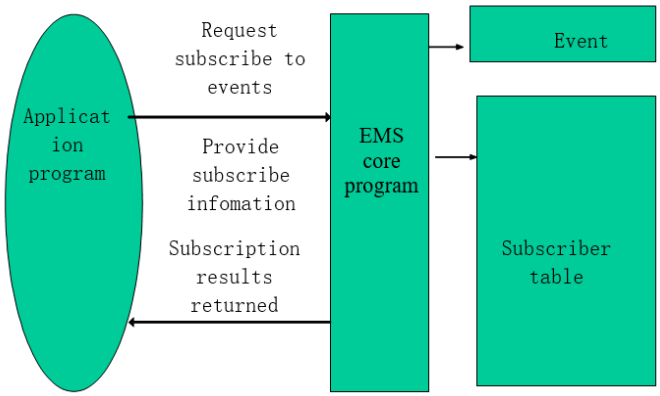

Figure 3. EMS event subscription model of messaging middleware

Application program procedures subscription request, it passed to the EMS core information includes: event name, subscription mode, their related information (including notification mode, if it is, then the notification mode).

EMS core received subscription request, it needs to check the event table to ensure that applications program to subscribe to the existence of such an event (event table is different from the event list, the former is used to mark the current system can provide the types of events, the latter The system has been received for the event); If the event does not exist, EMS will return failure.

EMS confirmed the event exists, but also check whether the application of some of the events strategy; If the event there is some strategy (such as allowing only certain features of the application program to subscribe to the event, with event-related strategy is identified when the

event publisher to register an event), it needs to be checked for subscribers to ensure that subscribers to meet these strategies, otherwise it returns failure. Through inspection, EMS core will create a subscriber entry, initialize it to the application program information provided, and then return success.

The above discussion, a situation not considered: only if the application subscription is a remote event. On the remote event subscription processing is the core of the local EMS as applied agency to request to the remote core EMS, the remote event receives a request, it will repeat the above steps, then register a subscriber to the subscriber table of local EMS items, the final result will return to the local EMS. The whole process may time out, the local EMS will be responsible for dealing with this time-out, and return an error to the subscriber.

Associated with the subscription process is event publishing process, the following Figure 4 summarizes the Message-Oriented Middleware EMS event publishing process:

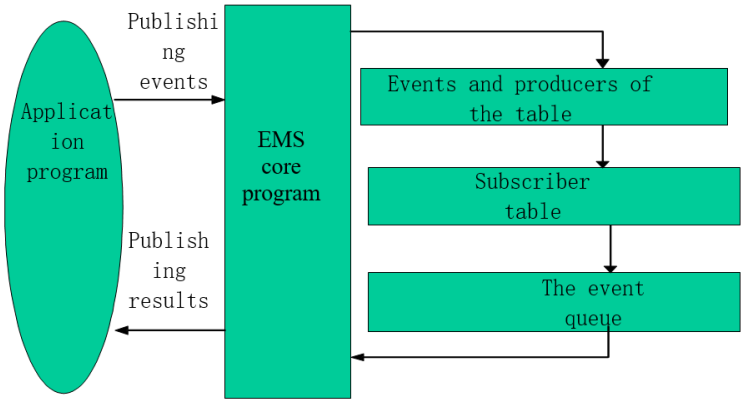

Figure 4. Message middleware EMS event model release

It can be seen from the figure, application program publishing events, the local Message-Oriented Middleware EMS first request; EMS receives event requests issued to the application, the first checks the event table and producer table to ensure effective release of the event; If the events related to strategies, which means that EMS also certify for released the event of an application program to ensure that it meets the events related to strategies.

EMS will then query the subscriber table by the event table; Subscriber table can find out information of application program such subscription events in the current; EMS use it to subscribe to the events of local or remote application notice; Between Event table and subscriber table relationships can be broadly described as: Each event table entries (on behalf of a class of events) has a pointer to subscribe to the list of subscribers to the event; For each type of event has a subscriber list, EMS informed through this list of those applications to subscribe to the current events.

If the EMS find a subscriber that it hopes to "View by" subscription event, the EMS will apply in the event list, a table entry into the event list, based on information provided by the subscriber or the system specified The default is to specify a life cycle event. When the list of

events in the event life cycle or the reference count to 0, this event will be removed from the list of events by EMS.

On a remote subscriber, the local EMS will report directly to the EMS where remote subscribers to be, the remote EMS notification by the remote subscribers.

Since the Message-Oriented Middleware EMS allows application-defined events, Message-Oriented Middleware EMS need to provide a mechanism to allow the application of registered events and logoff events. This process is similar to the WINDOWS user information registration process; Application by calling xxmidRegisterEvent () API request for registration, the EMS core is responsible for automatically assign a type number for new registration event (event type); Then the application can use this type number or the name of the event (provided in the registration event) for event publication and subscription.

Exists in the Message-Oriented Middleware EMS is an important question: how an application to know whether certain types of events are registered? Or how an application to obtain a certain type-related events information? The current Message-Oriented Middleware system can not finish a global notification be when an application registered event; In order to ensure the event type and name uniqueness, Message-Oriented Middleware EMS, requires users to register the event to ensure that such events (consistent with the user-specified name of the event) does not exist; This Message-Oriented Middleware EMS provides xxmidQueryEvent () API, the API requires name of the event, EMS will be based on the name to query for the entire system to obtain event-related information.

In addition, in order to ensure the uniqueness of the event type number, Message-Oriented Middleware EMS in the startup to the higher (Message-Oriented Middleware used in a tree structure) request the regional distribution of an event type, event type regional area can be free to use by this node Message-Oriented Middleware EMS;A parent node Message-Oriented Middleware EMS must ensure that this area should be divided not repeatedly. Current,Message-Oriented Middleware EMS have not yet to find a satisfactory solution; EMS on a node may returns an error when the application register the event, because the node can use the event type ID is limited (although there may be many event type ID has not been used).

Cancellation of the event is a reverse process. After the event is canceled, EMS will attempt to notify all application of subscribers the event, and then remove the same type of event-related resources. As the Message-Oriented Middleware EMS for event management, there are still these shortcomings, it is expected in future versions might consider using a distributed database to resolve use of the event type ID and event name.

2.3 EMS GSRA Algorithm

Message-Oriented Middleware EMS using GSRA Algorithm to collect the global state of the process represented briefly by the following Figure 5:

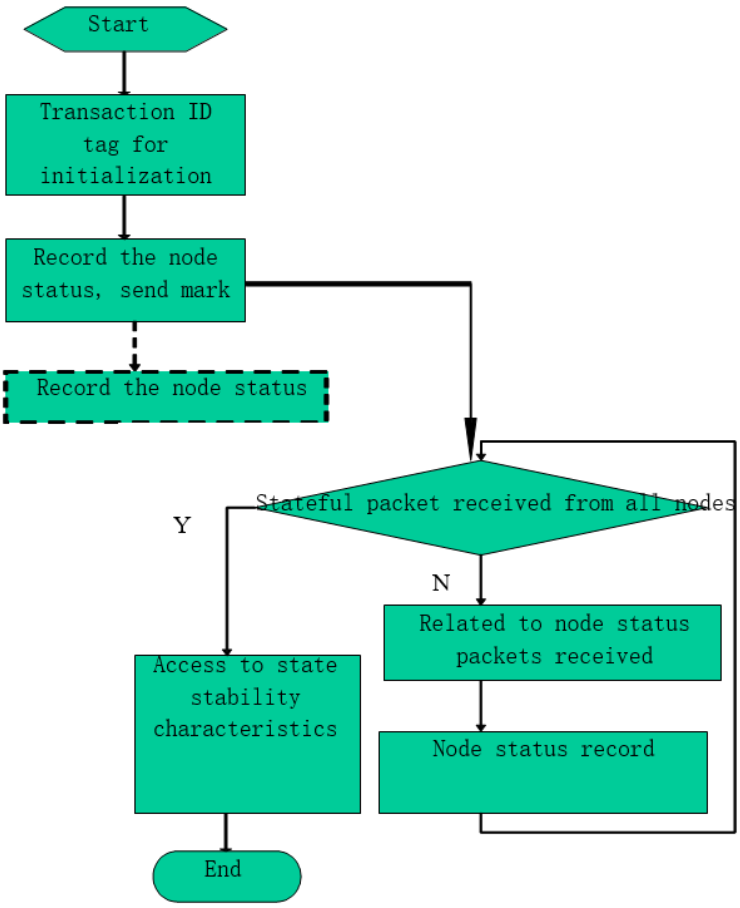

Figure 5. GSRA the process of collecting global state of sponsors

As the initiator of GSRA algorithm, in addition to run the code collection global status, it also needs to respond like other GSRA algorithms, record the status of their own nodes and connected to channel (figure dashed line);This recording process as shown Figure 6:

To point out that, despite the Message-Oriented Middleware EMS achieve GSRA algorithm, but its not currently being used; Main purpose using GSRA Algorithm is to support transaction processing may occur in the rollback operation.

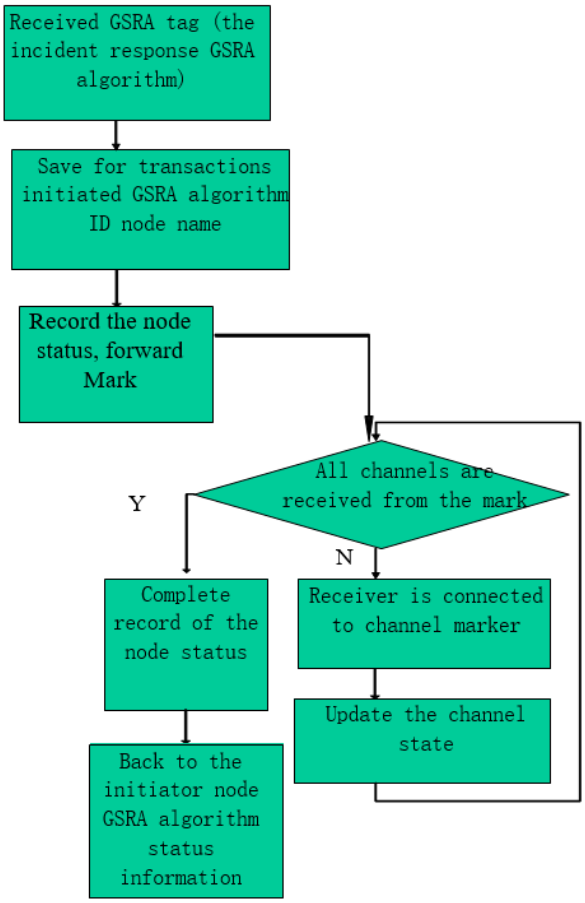

Figure 6. GSRA response to those processes (the node state record)

3. ANALYSIS OF TEST RESULTS

In order to study the feasibility and effectiveness of this technology, experiments were first conducted on Linux system servers. This server is composed of six IBM rack mounted servers. Firstly, when there is data input, event management is triggered, and the system prompts to generate public and private keys, as shown in Figure 7. Then check if the server generates public and private keys, as shown in Figure 8 and Figure 9.

According to the experiment, it can be concluded that the event management system of the message middleware system is completely feasible.

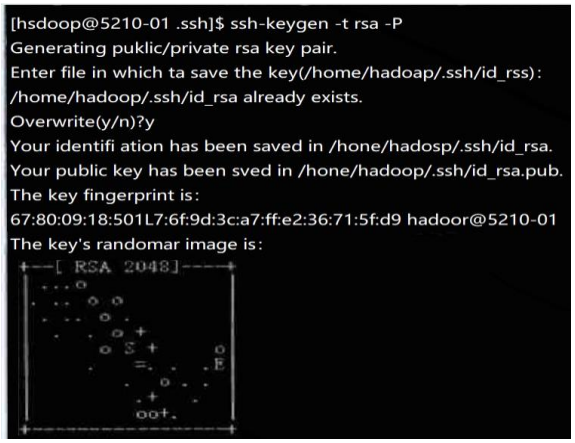

Figure 7. Generate public and private keys

Figure 8. Generate public keys

Figure 9. Generate private keys

4.CONCLUSION

Message-Oriented Middleware in distributed processing play an increasingly important role and in internet technology is currently a hot research[4].Development efficient, safe, reliable Message-Oriented Middleware suited to China's national conditions for the construction of China's information technology development is very important.Currently a number of middleware products have been widely applied to public security bureau, telecommunications, banking, securities and other important national sectors and industries.Some Message-Oriented Middleware products used in a number of technical and performance achieved or even exceeded, such as catch up with IBM, BEA and other companies of similar products.

Acknowledgment: First of all thanks to my mentor, Professor Miao Fang put the careful attention and guidance.Teacher Miao showing the teacher's rigorous scholarship and depth of expertise that I will never forget.Second, I want to thank the students living in and around my friends, they are my classmates Rui Liu, Ye Cheng-ming, they give me a lot of help, give me a great deal of relevant information and data.Finally, I would like to thank Beijing TongTech, they provided to me the papers design environment.

REFERENCES

[1] S. Cejka, A. Frischenschlager, M. Faschang, M. Stefan, and K. Diwold, " Operation of modular smart grid applications interacting through a distributed middleware," Open Journal of Big Data, vol. 4, no. 1, pp. 14-29, Jan. 2018.

[2] Chen Bingxing, Qiu Baozhi. Construction technologyofclustermessage-orientedmiddleware[J].ApplicationResearchofComputers, 2012(05):36- 38.

[3] Guo Xirong. Research on Digital Tourism Engineering and Evaluation Technology Based on G/S Model [D]. Chengdu: Chengdu University of Technology, 2002.

[4] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, " Middleware for Internet of things: A survey," IEEE Internet of Things Journal, vol. 3, no. 1, pp. 70-95, Feb. 2016.

[5] Zhang Yunyong, Zhang Zhijiang, Liu Jinde, et al. Principles and Applications of Middleware Technology [M] Beijing: Tsinghua University Press, 2004.

[6] Jammalamadaka R C , Gamboni R , Mehrotra S ,et al.A middleware approach for building secure network drives over untrusted internet data storage[J]. 2022.

[7] Hu Bin, Lin Zongkai, Guo Yuchai, et al. Design and Implementation of Multi Layer Architecture Middleware (Edb Client, EdbServer) [J] Computer Research and Development, 1998 (10): 870-872.

[8] S. Cejka, A. Frischenschlager, M. Faschang, and M. Stefan, " Security concepts in a distributed middleware for smart grid applications," in Proc. of the Symposium on Innovative Smart Grid Cybersecurity Solutions, Vienna, 2017, pp. 104-108.