International Journal of Grid Distribution Computing
Vol. 8, No.6, (2015), pp.193-202
http://dx.doi.org/10.14257/ijgdc.2015.8.6.19

Study of Concurrency Message Bus based on Adaptive Communication Environment Framework

Feng Ruan[1], Tao Li[1], Zhiyong Fan[1] and Jin Wang[2]

[1]*School of Information & Control,*
Nanjing University of Information Science & Technology, Nanjing 210044, China
[2]*College of Information Engineering, Yangzhou University, Yangzhou, China*

Abstract

Message middleware is an important branch of middleware, it provides the development of distributed application system with an asynchronous, pine for programming architecture, the consistent thought and the development trend of modern software technology, which is widely used in industry in the enterprise application integration and distributed computing. Message bus message middleware is a kind of simple implementation, but with its unique advantages for a wide variety of distributed applications development injected with strong power, greatly promote the development of application system integration. This paper is mainly focusing on the key issues involved in the middleware, message bus, and concurrency model for further research.

Keywords: *Adaptive communication environment, message bus, concurrency*

1. Introduction

Since the 1990s, the computing technology gradually came into the network which centered in the new period, the user was eager to build more abundant in the network distributed client/server applications. Not only realize data sharing, knowledge sharing and support and all kinds of computing resources sharing; it can realize at all levels, including the entire enterprise work together. In order to adapt to the demand, the distributed software system is rapidly becoming the concerned focus of researchers and users technology, and becoming a mainstream direction of the development of the computing technology.

Distributed software system is to support distributed processing software system that is connected by communication network perform tasks on a multiprocessor architecture system, contain an arbitrary number of system process and the user process, needs to implement a range of control in the system, in order to provide cooperation between the dynamic process and run time management. Compared with the traditional centralized computing system, distributed system showed the following advantages:

- Through parallel processing can improve the performance of the system.
- Through modular technology can improve the scalability of the system.
- Through the dynamic configuration and reconfiguration function can improve the system scalability.
- Through resource sharing can improve the system performance/cost ratio.
- Client and server FGC, easy to implement layered and abstract.

ISSN: 2005-4262 IJGDC

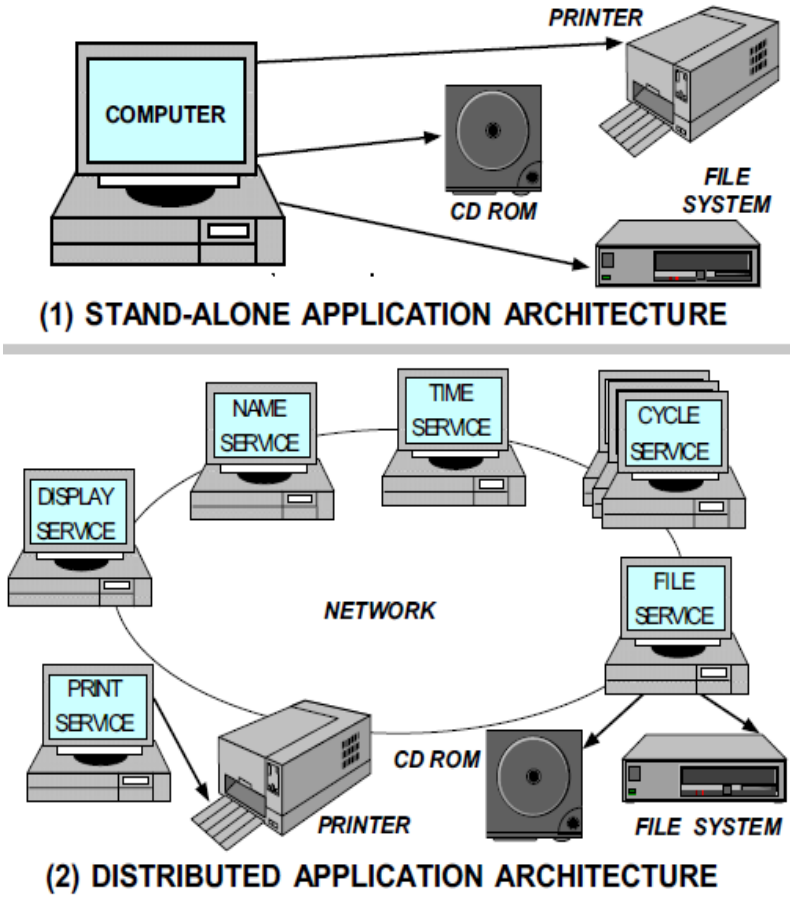

Figure 1. Motivation for Distribution

In a distributed system, in order to solve the large-scale application system contradiction between integrity and extensibility, there come middleware technology application. Figure 1, shows the motivation for distribution. Middleware is a software agent in the application layer and network layer between a function level, independent of the application system by heterogeneous operating systems, hardware platform and communication protocol of the underlying environment. The middleware has become a key foundation of distributed computing software, the middleware, can better development, operation and management of distributed application system. The development of middleware technology has been increasingly mature, and the different levels and different types of middleware products. Middleware is usually divided into six categories, respectively is message oriented middleware, transaction middleware, data access middleware, object middleware, remote procedure call middleware and terminal access/shielding conversion middleware.

The Adaptive Communication Environment (ACE) is an object-oriented (OO) toolkit that implements fundamental design patterns for communication software. ACE is targeted for developers of high-performance communication services and applications on UNIX and Win32 platforms. ACE simplifies the development of OO network applications and services that utilize intercrosses communication, event demultiplexing, explicit dynamic linking, and concurrency. ACE automates system configuration and reconfiguration by dynamically linking services into applications at run-time and executing these services in one or more processes or threads [1].

Certain types of distributed applications benefit from using a concurrent model of execution to perform their tasks. Concurrency is particularly useful to improve performance and simplifying programming for network servers on multiprocessor

platforms. For server applications, using threads to handle multiple client requests concurrently is often more convenient and less error-prone than the following design alternatives: Artificially serializing requests at a transport layer interface, queuing requests internally and handling them iteratively, and forking a heavy-weight process for each client request [2,3].

ACE encapsulates and enhances the lightweight concurrency mechanisms provided both by Solaris 2.x threads [4], POSIX Pthreads [5], and Win32 threads [6]. The material presented in this paper is intended for a technical audience interested in understanding the strategies and tactics of object-oriented (OO) concurrent programming using threads. It is assumed that the reader is familiar with general OO design and programming techniques (such as design patterns [7], application frameworks [8], modularity, information hiding, and object modeling [9]), OO notations (such as OMT [10]), fundamental C++ programming language features (such as classes, inheritance, dynamic binding, and parameterized types [11]), basic UNIX systems programming concepts (such as process management, virtual memory, and inter-process communication [12]), and networking terminology (such as client/server architectures [13], RPC [14], CORBA [15], and TCP/IP [16, 17]).

Organization of the paper: The main content of this paper is constructed in 6 sections as follows: Section 2 gives an overview about advantages of ACE. Section 3 describes the middleware. Section 4 describes message bus. Section 5 discusses the concurrency of ACE. Section 6 concludes our work and future work.

2. Advantages of ACE

ACE is not just a class library. It is a powerful, object-oriented application tool kit. In order to separation of concerns, reducing complexity, allowing the division of function modules, the ACE tool kit design USES a layered architecture, is the ACE framework architecture diagram in Figure 2.The framework consists of the following components

A. OS Adaptation layer

The layer directly resides on local OS API written in C. It provides lightweight class POSIX OS adaptation layer, layer and other components in the ACE and the following platform proprietary features associated with OS API shielding:

- Adaptation layer encapsulates the concurrency and synchronization: ACE for multithreading, multi-process and synchronous OS API, inter-process communication (IPC) and Shared memory. The ACE of the adaptation layer encapsulates for local and remote protection OS API C, and Shared memory.
- Event multiplex separation mechanism: the ACE of the adaptation layer encapsulates for based on FO, timer, and signal synchronous and asynchronous and synchronous events multiple separate OS API.
- Explicit dynamic link: the ACE of the adaptation layer encapsulates the OS API for explicit dynamic linking. Explicit dynamic link allows the installation or run-time configuration for application services.
- File system mechanism: the ACE of the adaptation layer encapsulates the for manipulating files and directories of the OS file system API.

B. C++ The packing layer appearance

- ACE package provides a number of C++ the same features as the ACE OS adaptation layer. However, these features is to use C++ classes and objects, rather than an independent C function to structure, including the concurrency and synchronization component, the IPC and the file system components, memory management module, *etc*.

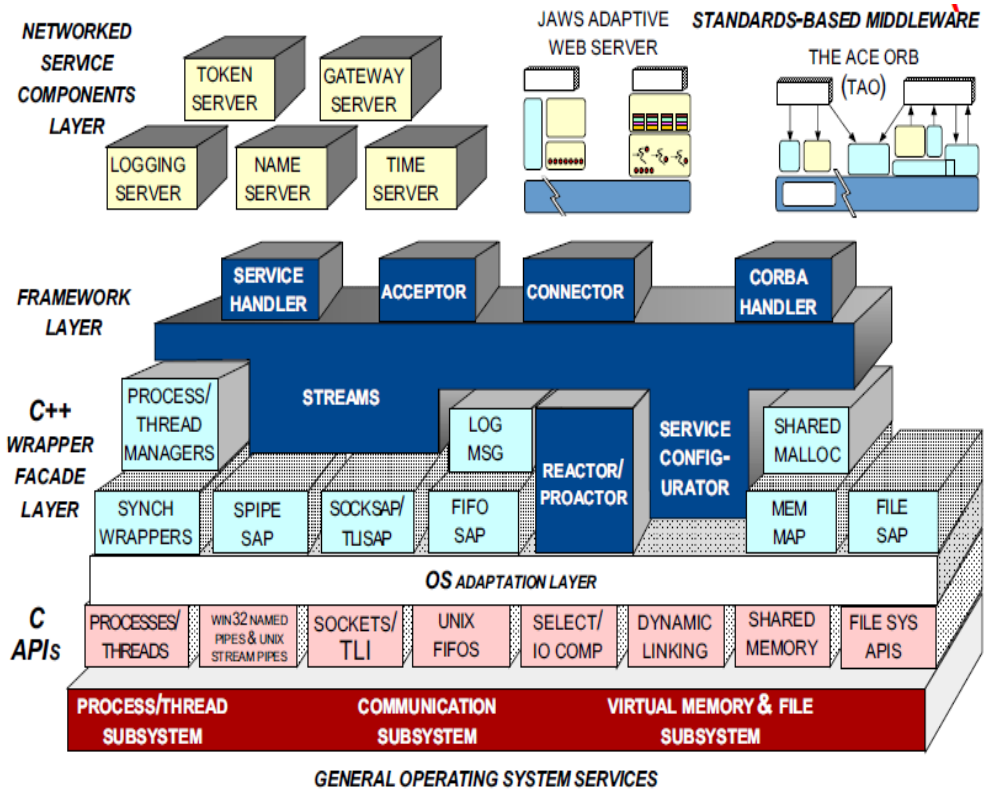

Figure 2. ACE Framework Architecture Diagram

C. The Framework Layer

ACE also contains an advanced network programming framework, the integration and enhances the lower level C material packaging appearance. The framework supports concurrent distributed service dynamic configuration into the application. ACE framework section contains the following components:

- Event multiple separate components: ACE Reactor (reactor) and Pro-actor (proactive) is extensible object-oriented multiple separator, their proprietary dispatching application processor, in response to a variety of types based on FO, timer, signal and synchronization event.
- Service initialization module: ACE Acceptor (receiver) and connector (connector) initialization tasks respectively to make active and passive components and initialize once after the completion of the communication services on the application of proprietary mission to harm us.
- Service configuration module: ACE Service Configurator (service) support application configuration, these applications services in the installation and/or run-time dynamic assembly.
- Stratified flow components: ACE stream component simplifies like user-level protocol stack is composed of layered services for the development of communication software application.

D. Network Services and Components

Provide complete, reusable services, including configuration service, naming service and the log service, *etc*.

ACE framework design purpose is to provide a portable, high performance and highly object-oriented application development kit, promote the development of high-performance, real-time distributed object computing framework. So ACE in design full consideration to the network communication program in facing problems, and using

patterns and object oriented method, provides an elegant solution. The use of ACE toolkit for developers to bring the following advantages:

A. Simplify the complexity of network programming

Need developers to consider network time delay in network programming, byte sequence, the data structure layout and network instability and performance problems, ACE provides many excellent implementation framework and model, to encapsulate these tumultuous programming details, allowing developers to liberated from complex underlying communication processing, better focus on the development of the upper application.

B. Provides a safe and easy to use interface encapsulation, reduce the error rate

Most APIs are provided by the system in accordance with the written CAPI, CAPI is a kind of weak type of low-level APIs, and different platform for the implementation of these APIs and presentation is different, so have brought difficulties for development and debugging. ACE using strongly typed C++ language implementation, encapsulate low-level API for different platforms, provides a safe and easy to use interface for users, effectively reduce the probability of programming errors.

C. Provides a rich framework, patterns, and components

Improve the development efficiency of ACE contains some commonly used in web development framework, patterns, and components, to achieve the specification control flow and the object of collaboration, thus provides a version of the completed application. The framework gives developers the ability of large-scale reuse software, simplifying the based on the complexity of the framework to build applications.

D. Portability is strong

The ACE can be run on most PCs, such as Windows, Linux, Debian, *etc*., can also be run on most UNIX systems, such as Sunos4. X and Solaris HP - UX, Digital UNIX (CompaqTru64), AIX can run on most real-time operating system。

E. Open source

ACE is open source software for free personal research. Enterprise can be in the premise of comply with the GPL to use ACE framework to build business applications.

3. Middleware

Middleware is an independent software layer, it provides the platform and system between the universal services, has a standard program interfaces and protocols, avoiding the application system and concrete platform between the FGC tight. Middleware lies between system layer and application layer, it is down to block the difference of hardware platform or operating system platform, up to the application layer provides a unified operation program interfaces and protocols, application development based on the interface, the realization of different hardware and operating system platform application data sharing and interoperability. On the specific implementation, the middleware is distributed software frameworks are defined with the API, with strong communication ability and good extensibility. The difference between the middleware and application software is: middleware design fully consider the generality, and provides a standardized programming interface API, can be invoked and secondary development software.

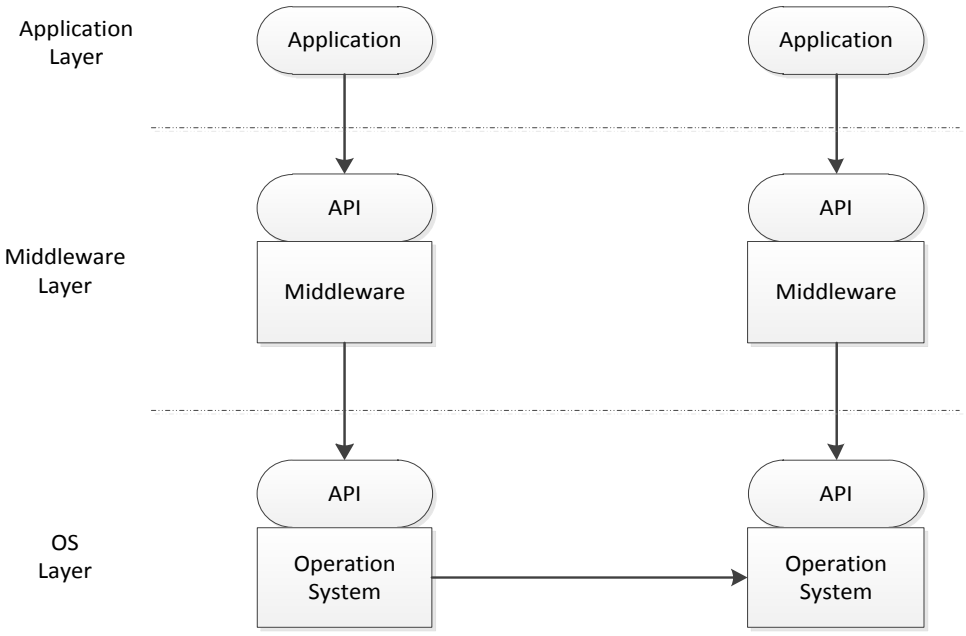

Figure 3. Model of Middleware Application System

Message oriented middleware mainly through messaging to complete the processing of data and control distributed computing environment. It provides a complete processing environment, allows developers and users to connect between different systems of data and code, or a consistent interface for application processing of interconnection.

From the vertical perspective, the message middleware functions well in connecting the user application and the basic communication mechanism. From the developer point of view, message middleware is good encapsulate different distributed environment, provides a unified interface to make the application and business can collaborate with unified interface. Message oriented middleware's function mainly includes: the message queue, the trigger function, send messages, security control, data broadcast, based on the cost of routing and debugging and management functions, *etc*.

Message middleware are common in establishing large-scale distributed applications, is the main mode of communication radio/ordering (the Publish - Subscribe) and point to point (peer - to - peer) way. Based on message middleware application does not need to (or network) and communication media and the coupling between the remote applications do not need to consider the complexity of the communication between distributed applications, from the physical connection to the application, liberated from the complex underlying communication better focus on business.

4. Message Bus

Message transmission bus mobile intelligent network is mainly oriented to support system in the distributed system based on message communication, committed to providing an efficient and powerful communication framework, making the development of the business layer can be liberated from the depth of the complex network communication. From the point of practical application, system often USES a manager/agent model, such as the MS system based on TMN model, in the application layer polling processing strategy based on a certain time granularity and request/response mode of communication, so communication with concentration, sudden. And in recent years, with the gradual improvement of the management of mobile business system requirements, system network scale is more and more big but system USES time granularity is more and more small, it's on the efficiency of the underlying

communication components and performance put forward higher requirements.

Therefore, the system in the design must follow the following principles:

- High performance. This is decided by the system of the upper application characteristics and system structure. System is controlled by a central message processing several distributed server and the client, so when the system is very large, the central message processing server processing performance must be very high. And, as a result of the system in the existing applications show the message flow is a kind of broken hair message flow of relatively concentrated, so the system must have a high processing efficiency, to ensure that messages on the server without blocking or lost.
- High reliability. Due to the system as a carrier class support to the underlying communication components in the system, so it must be the reliability of the telecommunication level and ensure 7 * 24 hours of continuous operation, in the architecture design shall consider the system as whole disaster ability, ensure the system run reliably.
- High extensibility. System module design needs to consider the versatility and extensibility, the public function module componentization, increase software reusability. On the one hand, the client API must support the comprehensive communication control ability, to provide users with a variety of ways of call interface; The server side, on the other hand, USES the object-oriented idea and excellent model to architecture, makes the individual components and processing can be modular, loosely coupled, especially when adding new message processing tasks using plug-in.
- High maintainability. System with high maintainability, provide the operation and maintenance of command and access method, can be convenient to online system maintenance and upgrade. At the same time, the system server need to implement some management functions, such as real-time monitoring of the message flow and message persistence services, *etc*.
- Compatibility with existing products. The purpose of the system design is to provide a high performance communication service bus, replace the existent MRB components, so must be considered in the design and your own product compatibility problems, through the programming to an interface mechanism implementation with your own the seamless integration of the system.

5. Concurrency

5.1. Overall Requirements

In conjunction with the goal of encapsulating and simplifying the concurrency substrate of OS threading mechanisms, the ACE OO thread encapsulation class library is being developed in response to the following common application requirements.

- Simplify program design – by allowing multiple application tasks to proceed independently using conventional synchronous programming abstractions (such as CORBA remote method invocations);
- Transparently improve performance – by using the parallel processing capabilities of hardware platforms such as the SPARC center 1000 and 2000 shared memory symmetric multi-processors;
- Explicitly improve performance – by reducing data copying and by overlapping computation with communication;

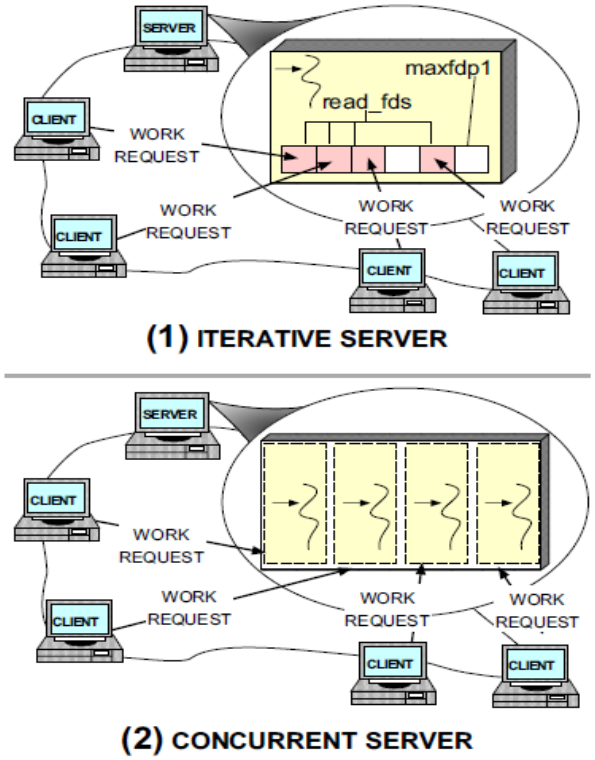

Figure 4. Motivation for Concurrency

- Improve perceived response time – for interactive applications (such as user interfaces or network management applications) by associating separate threads with different tasks or services in an application.

5. 2 Benefits of Threads-based Concurrent Programming

It is often advantageous to implement concurrent applications that perform multiple tasks in separate threads rather than in separate processes for the following reasons:

- Thread creation – unlike forking a new processes, spawning a new thread does not require (1) duplicating the parent's address space memory, (2) setting up new kernel data structures, and (3) consuming an extra process slot in order to perform a subtask within a larger application.
- Context switching – Threads maintain minimal state information. Therefore, context switching overhead is reduced since less state information must be stored and retrieved. In particular, context switching between threads is less time consuming than context switching between UNIX heavyweight processes. This is due to the fact that TLB virtual address mappings not need be.

6. Conclusions

Message middleware is an important branch of middleware, it provides the development of distributed application system of an asynchronous and low FGC programming architecture, the consistent thought and the development trend of modern software technology, which is widely used in industry in the enterprise application integration and distributed computing. Message transmission bus based on ACE framework of the implementation of a message bus, mobile intelligent network is mainly oriented to support system in distributed systems based on message communication, committed to providing an efficient and powerful communication framework, making the

development of the business layer can be liberated from the depth of the complex network communication.

Acknowledgments

This paper is a revised and expanded version of a paper entitled "Research of Message Bus Middleware based on ACE" presented at COMCOMS 2015, Hanoi, Vietnam, October 22-24, 2015. This research work was supported by the National Natural Science Foundation of China (61402234), and the Industrial Strategic Technology Development Program (10041740) funded by the Ministry of Trade, Industry and Energy (MOTIE) Korea.

References

[1] D. C. Schmidt, "An Object-Oriented Network Programming Toolkit for Developing Communication Software".
[2] D. C. Schmidt, "An OO Encapsulation of Lightweight OS Concurrency Mechanisms in the ACE Toolkit"
[3] D. C. Schmidt, "ACE: an Object-Oriented Framework for Developing Distributed Applications," in Proceedings of the 6th USENIX C++ Technical Conference, (Cambridge, Massachusetts), USENIX Association, April **(1994)**.
[4] J. Eykholt, S. Kleiman, S. Barton, R. Faulkner, A. Shivalingiah, M. Smith, D. Stein, J. Voll, M. Weeks, and D.Williams, "Beyond Multiprocessing... Multithreading the SunOS Kernel," in Proceedings of the Summer USENIX Conference, (San Antonio, Texas), June **(1992)**.
[5] IEEE, Threads Extension for Portable Operating Systems (Draft 10), February **(1996)**.
[6] H. Custer, Inside Windows NT. Redmond, Washington: Microsoft Press, **(1993).**
[7] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, "Design Patterns: Elements of Reusable Object-Oriented Software. Reading", MA: Addison-Wesley, **(1995)**.
[8] R. Johnson and B. Foote, "Designing Reusable Classes," Journal of Object-Oriented Programming, vol. 1, pp. 22–35, June/July **(1988)**.
[9] G. Booch, "Object Oriented Analysis and Design with Applications (2nd Edition)", Redwood City, California: Benjamin/ Cummings, (**1993**).
[10] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, "Object-Oriented Modeling and Design", Englewood Cliffs, NJ: Prentice Hall, **(1991).**
[11] Bjarne Stroustrup, The C++ Programming Language, 2nd Edition. Addison-Wesley, (**1991**)
[12] W. R. Stevens, "Advanced Programming in the UNIX Environment", Reading, Massachusetts: Addison Wesley, **(1992).**
[13] D. E. Comer and D. L. Stevens, "Internetworking with TCP/IP Vol III: Client – Server Programming and Applications", Englewood Cliffs, NJ: Prentice Hall, **(1992).**
[14] Sun Microsystems, "Open Network Computing: Transport Independent RPC", June **(1995).**
[15] Object Management Group, "The Common Object Request Broker: Architecture and Specification", 1.2 ed., **(1993)**.
[16] W. R. Stevens, UNIX Network Programming, First Edition. Englewood Cliffs, NJ: Prentice Hall, **(1990)**.
[17] W. R. Stevens, TCP/IP Illustrated, Volume 1. Reading, Massachusetts: Addison Wesley, **(1993)**.
[18] J. Wang, J-Uk Kim, L. Shu, Y. Niu and S. Lee, "A distance-based energy aware routing algorithm for wireless sensor networks", Sensors, 10, 10, **(2000)**.
[19] J. Wang, Y. Yin, J. Zhang, S. Lee, and R. Simon Sherratt, "Mobility based energy efficient and multi-sink algorithms for consumer home networks", IEEE Transactions on Consumer Electronics, 59, 1, **(2013)**.

Authors

Feng Ruan, lecturer at Nanjing University of Information Science & Technology. He received the B.S. and M.S. degree in the Nanjing University of Information Science & Technology. Now he is PHD student in Computer Science Department of Nanjing University of Information Science & Technology. His research interests mainly include complex system, computer network, routing protocol and algorithm design, data fusion, and cloud computing.

Zhiyong Fan received MSc from Nanjing University of Information Science and Technology (Nuist) in 2007, China. He is a lecturer in the School of Information and Control Engineering at Nuist, China. Now, he is a PhD student in Nanjing University of Science and Technology, China. His current research interests include medical imaging, image processing and pattern recognition.

Jin Wang, Professor at College of Information Engineering, Yangzhou University, Yangzhou, China. His research interests mainly include routing protocol and algorithm design, performance evaluation and optimization for wireless ad hoc and sensor networks. He is a member of the IEEE and ACM.