IEICE TRANS. INF. & SYST., VOL.E87–D, NO.2 FEBRUARY 2004

LETTER

A Flexible Distributed Computing System and Its Application for Signal Processing

Zhihui WANG[†], *Nonmember*, **Tohru KIRYU**[†], *Member*, **Keisuke SHIBAI**[†], *and* **Shinkichi SAKAHASHI**[†], *Nonmembers*

SUMMARY In this paper, we present a flexible distributed computing system in which it is very easy to add required computing components at any time. The system is an Internet-based solution, and mainly developed by Java and XML. Moreover, by implementing a new configuration of computing information that is setting up Public Information and Private Information, the system can accommodate various computing requests, and facilitate a flexible design. Additionally, to show the practical merit, as an example of signal processing, we presented how to apply our proposed system to selection of a suitable artificial neural network.

key words: *distributed computing, Java, XML, management flexibility*

1. Introduction

Distributed computing works by dividing large tasks into many smaller ones, and distributing them to some other computing devices to realize computing simultaneously via a network, such as a private corporate network or the Internet. After the tasks are processed with individual computing devices, the results are transmitted back to a central server that subsequently assembles an answer. In this case, as software applications become large and complex, distributed computing technology can help to constitute sophisticated environments [8].

A number of studies have been conducted on the cooperative use of distributed resources that unify to solve signal processing problems. Among those studies, SETI@home project [9] is the first attempt and the most famous research project to use large-scale distributed computing to perform a sensitive search for radio signals from extraterrestrial civilizations. However, such studies focused on improving the speed of a specific signal processing algorithm. They ignored the fact that every computing unit has enormous independent development ability. Trying to extend the developing capability of each computing unit, and offering as many signal processing solutions in a virtual organization as possible, should be an important point that would need to be settled for Internet-based distributed computing in the signal processing field. On the other hand, some studies provided collaborative computing environment [4], [5], but these studies didn't implement a flexibility issue.

At the point of improving computing speed, general parallel computing can be used to solve a specific problem. On completing one execution, each task waits at a joining point for its sibling tasks to complete execution, which requires a synchronous process [2], [3]. However, the core control source codes must be rewritten in case of parallel computing if requests of the problem is changed for some reasons, for example, the inner structure of the problem has to be modified according to different conditions, or it needs to compare several respects of the problem at the same time. Thus parallel computing is not suitable for treating such appliances. By distributed different tasks or different structures of one task to distributed computers, it is available to effectively deal with such problems under an asynchronous way.

In this paper, we present a flexible distributed computing (FDC) system that sets arbitrary numbers of self-configuring distributed objects to help solve some signal processing problems. At present, no single organization can offer all solutions in signal analysis. The cooperation of organizations is an inevitable trend because it can greatly improve the ability to solve signal processing problems. To accommodate such a cooperative environment, we designed some methods that gave the FDC system flexible characteristics, such as easily adding new signal processing algorithms and offering each organization plenty of independent development ability.

2. Methods

We adopted a three-tier Client/Server architecture as a distributed computing infrastructure model. The mul-titiered structure has been employed to solve some problems of client- and server-side processing, which makes system upgrades difficult [1]. The three tiers comprise a front-end tier, a middle tier, and a back-end tier. Like typical structures, the front-end tier is a client application tier and resides on the end-user computers. The middle tier is a central server, consisting of web servers to serve up static contents, and application servers to serve up dynamic contents. Unlike typical structures, the back-end tier is a distributed object server, not a database server. This model doesn't use any database, but only realizes a small data storage application.

Figure 1 presents the system implementation that supports the detailed technologies and processing flow for the FDC system. The central server takes charge of receiving and transmitting user data, logic analysis and dealing with querying results. The communication mechanism between the central server and the client side consists of TCP sockets to handle requests to retransmit missing packets.

Manuscript received October 4, 2002.
Manuscript revised July 8, 2003.

[†]The authors are with the Graduate School of Science and Technology, Niigata University, Niigata-shi, 950-2181 Japan.

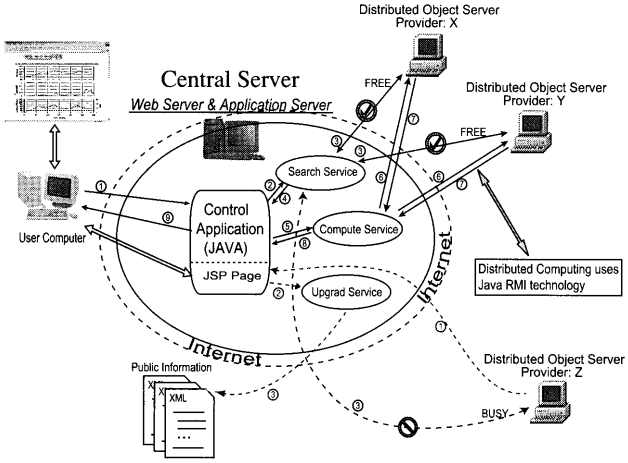

Fig. 1 FDC system implementation.

The central server provides several services, which are at the bottom level of the FDC system, to deal with higher-level concepts such as searching, computing and upgrading. It is easy to join others services developed by the other organizations to the central server. Moreover, it is also easy to change the availability of a service when its capacity becomes limited. The searching service is responsible for finding suitable distributed object servers to carry out computing tasks based on complexity of current work and executive statuses of object servers. Every distributed computing task should be started from requesting a searching service. Certainly, computing service is the kernel of computing component in the FDC system, and will treat all real computing operations. Practical computing is performed in the distributed object server tier under control with the computing service, and it is responsible for the distributed object servers to guarantee the computing methods' validity and quality. An upgrading service, which is responsible for making system automatically upgradable, presents the FDC system as a flexible one.

As our distributed computing solution, we selected Java Remote Method Invocation (RMI) technology [6] because it is the basis for distributed computing in the Java environment. Moreover, we extended the FDC system so that it could use another language in some situations, especially when creating real computing algorithms. In practical, we used XML (eXtensible Markup Language) [7] to manage the stated information of various types methods. XML language offers data portability and reusability across different platforms and devices.

The information from the central server and all distributed object servers are preserved in an information library of the central server, which complies with XML standards. Besides, each distributed object server also holds its proprietary configuration information. Indeed, the use of XML as interfaces of each computing unit makes integration easy, and accordingly, improves flexibility of the FDC system.

There are two kinds of information in the information library: Public Information and Private Information. Public Information is used for multi-distributed computing and is visible on all distributed object servers, whereas Private Information is only visible on one specific object server, which provided this information. In order to accommodate this kind of design, the upgrading service in the central server provides two upgrading methods: upgrading Public Information, named the Whole System Upgrade; upgrading Private Information, named the Individual Object Upgrade. Individual Object Upgrade, in particular, realizes automation from viewpoint of the central server, and also gives sufficient extensibility to each distributed object server for enhancing its process ability. The distributed object server, which wants to update its private computing ability (i.e. it has developed some new methods), just overwrites its XML-based compatible computing configuration existing on the central server. On the other hand, updating the ability of multi-distributed computing is executed on the central server and all actual processing algorithms will be revised on each distributed object server. Note that a task-dividing method, which includes how to divide one whole work into different parts and does not influence the accuracy of other results, is a basic process of distributed computing technology. All task-dividing methods are provided with distributed object servers in case of the FDC system.

3. Experiments and Results

We have accomplished an example that involved selecting an appropriate Artificial Neural Network. The task took a correspondingly long time if computed in one local computer, and it was not suitable to be processed by general parallel computing.

The experiment was conducted on our research laboratory's intranet, which has 100 Base-T Ethernet connections. The central server and each distributed object server were all set dynamic IP addresses. We used two types of test schemes for the experiment. One was executing the computing task in one local advanced computer (Pentium-III 1.7 GHZ, 1 GB RAM) in the normal way. The other test scheme used the FDC system to perform a computing task with several generic computers (Pentium-II 400-500 MHZ, 128-256 MB RAM) to perform computing tasks. All of the experiments were implemented in the same software and hardware conditions. In addition, we just adopted a very simple task-dividing method in our experiment.

3.1 Selection of Artificial Neural Network

We have used an Artificial Neural Network (ANN) for developing the Internet-based cycle ergometer system, which services a personal fitting process for the elderly [10]. A suitable ANN should be designed for estimating the rating of perceived execution [11] from several objective indices (heart rate, workload, etc.). We designed several different ANN structures (different initial value and different number of units in the hidden layer), and selected a suitable one by

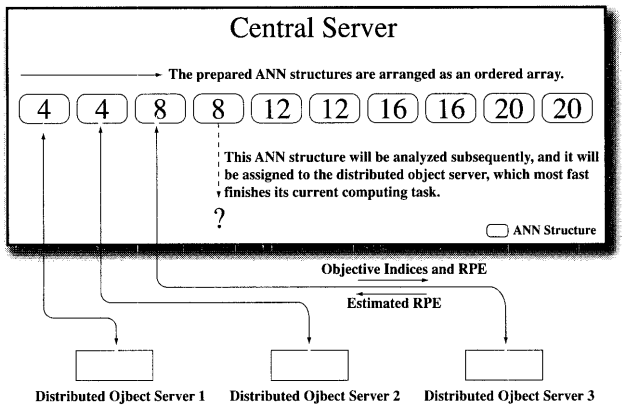

Fig. 2 An actual implementation method. The number of computing sets is 2, and the number of distributed object servers is 3.

the FDC system.

Our task-dividing method is briefly demonstrated as follows: all prepared ANN structures were arranged as an array in order at first. The number of sub-tasks equaled to the number of ANN structures. Therefore, each sub-task was responsible for one ANN structure, which was assigned to one distributed object server to perform actual analysis. Based on the conditions of computing environment (how many distributed computing servers were available.), if there were ANN structures remained to be analyzed, the ANN structure would be assigned to the distributed object server, which most fast finished its current computing task.

Figure 2 illustrates an example of actual implementation that the number of computing sets was 2, and the number of distributed object servers was 3. In our task-dividing method, since the set number was 2, the ANN structures were firstly arranged as an ordered array as follows: 4, 4, 8, 8, 12, 12, 16, 16, 20, and 20. Because only 3 computers were available for distributed object servers, the ANN structures, 4, 4, and 8, were dispatched to different object servers at the beginning of actual ANN computing. Afterwards, remained ANN structures waited for the release of one distributed object server. If one object server completed its current computing task, the next ANN structure, 8, in this example, was assigned to the available object server. Such dispatching work finished when all ANN structures were analyzed based on the ordered array.

In our experiments, input data were objective indices, which included the heart rate, a muscular fatigue index and other workload factors. Output data were estimated rating of perceived execution. We selected an ANN that showed the least squares of estimation error. To compare the elapsed time, we fixed three parameters: "number of hidden layer's unit", "training epochs", and "precision limit". These parameters were all set to default values that met most operation's needs. As an initial parameter, we only changed the "number of computing sets" in this experiment. Table 1 shows the details of the control parameters.

Table 1 Control parameters.

Same Parameters	number of hidden layer's unit: 4, 8, 12, 16 and 20; training epochs: 1000; precision limit: 0.005;
Different Parameters	number of computing sets (1 ~ 5);

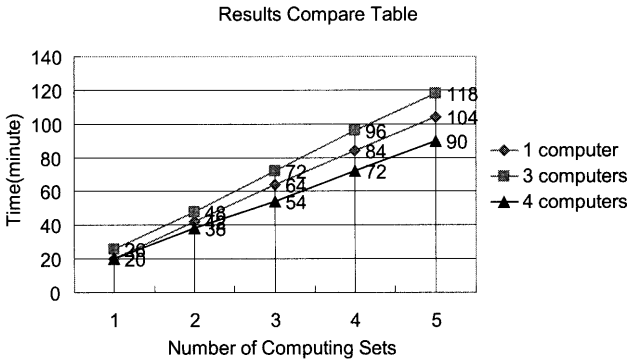

Fig. 3 Results comparison of Neural Network computing experiment.

3.2 Results

Figure 3 shows the comparison between the elapsed times of the two test schemas. Moreover, for the test schema using the FDC system, we carried out two experiments that were computed with three computers and with four computers. Actually, we also evaluated flexibility of the FDC system at the experiments when appending another computers, by only modified the Public Information on the central server.

In Fig. 3, the middle curve is the result of using the advanced computer. Obviously, computing speed of using four computers together was faster than the case of using the advanced computer. Although these curves were all similar to straight lines, the result of using this system was not always presenting a straight line except the case of using only one local computer. If we added another distributed object servers with greater or lower performance, or changed the task-dividing method, the curve would have evident turn points due to different computing abilities of object servers.

4. Discussion

Signal processing technology is being used extensively in various kinds of research fields. However, some problems are often very numerically and/or data intensive and consequently requiring a variety of heterogeneous resources that are not available on a local place [1]. Moreover, new analysis methods appear all the while. That means only one organization or one signal processing solution is insufficient to solve the many different complex problems that are appearing now. Unions of different organizations providing various kinds of methods together will be the trend for future developments in signal processing. The rapid growth of the Internet is offering this chance.

We presented a FDC system that implemented a computing approach for integrating distributed resources, and it adapted to a platform-independent manner by using Java and XML. For our experiments, the application for selecting a suitable ANN structures was presented to design different aspects of the computing task, and computing methods for different structures were diverse. In this case, general parallel computing approach is not a good choice, although it could complete our experiments by hardly coding scheduling algorithms and controlling methods [2], [3]. Easily adding new components and new analysis methods are an important point for the FDC system. According to different signal characteristics, task-dividing methods are different from each other. No matter what kind of signal it is, the FDC system can be designed to improve the speed of signal processing by using multi-distributed computing based on the task-dividing method provided in Public Information. Even if one computing task can't be divided to take multi-distributed computing, it is also possible to complete computing task on only one computer (one distributed object server in our case) by using the FDC system due to the Private Information.

There is still a lot of work that has to be done on the FDC system. For example, at the central server end, it needs realizing a real database operation to manage the information of users and distributed object servers, which will be constantly increased. In fact, we have implemented a simple agent in the central server for detecting performance and monitoring transferring traffic. For a future design, a more advanced software agent will be developed and be separated from the center server. In addition, at present, if one distributed server wants to join the FDC system, it is validated through the artificial authentication of the central server. Consequently, an automatic authentication should be realized.

5. Conclusions

With development of the Internet, ways to take full advantage of its abilities has become a significant problem. This study has shown the great possibility of long-time or multivariate signals analysis by using distributed computing technology and employing the Internet as a transmitting structure. We have developed a flexible distributed computing system that implemented our proposed solutions. Our designed computing model has been evaluated for selecting a suitable artificial neural network from many different structures. Furthermore, the flexible distributed computing system provides flexibility and independent platform ability.

References

[1] N.H. Lovell, F. Magrabi, B.G. Celler, K. Huynh, and H. Garsden, "Web-based acquisition, storage, and retrieval of biomedical signals," IEEE Eng. Med. Biol. Mag., vol.20, no.3, pp.38–44, May 2001.
[2] E. Varki, "Response time analysis of parallel computer and storage system," IEEE Trans. Parallel Distrib. Syst., vol.12, no.12, pp.1146–1161, 2001.
[3] H. Topcuoglu, S. Hariri, and M. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," IEEE Trans. Parallel Distrib. Syst., vol.13, no.3, pp.260–274, 2002.
[4] C. Lee, C. Chiang, and M. Horng, "Collaborative web computing environment: An infrastructure for scientific computation," IEEE Internet Comput., vol.4, no.2, pp.27–35, 2000.
[5] S. Kim, M. Hyun, and J. Yamakita, "FT_HORB: A fault-tolerant distributed programming environment based on RMI," IEICE Trans. Inf. & Syst., vol.E85-D, no.3, pp.510–517, March 2002.
[6] http://java.sun.com/products/jdk/rmi/
[7] http://www.xml.org/
[8] S. Park, J. Kim, and S. Lee, "Agent-oriented software modeling with UML approach," IEICE Trans. Inf. & Syst., vol.E84-D, no.4, pp.465–476, April 2001.
[9] http://setiathome.ssl.berkeley.edu/
[10] T. Kiryu, K. Shibai, Y. Maruyama, Y. Hayashi, and K. Tanaka, "Providing appropriate exercise levels for the elderly," IEEE Eng. Med. Biol. Mag., vol.20, no.6, pp.116–124, 2001.
[11] G. Borg, "Perceived exertion: A note on 'history' and methods," Med. Sci. Sports Exerc., vol.5, pp.90–93, 1973.