Application of Distributed Object Technology to Seismic Waveform Distribution

N. Takeuchi, S. Watada
Earthquake Research Institute, University of Tokyo

S. Tsuboi
Yokohama City University

Y. Fukao
Earthquake Research Institute, University of Tokyo

M. Kobayashi, Y. Matsuzaki, and T. Nakashima
Fujitsu Limited

INTRODUCTION

We develop a networked data distribution system for broadband seismic waveform data using Java RMI (Remote Method Invocation). Users can download data from various networks using a unified interface. There are two technical problems in building such a user-friendly system with a high capability for expansion: communication through firewalls and information synchronization between request interfaces. We solve these two problems using cgi (common gateway interface) filtering and the Directory Service, respectively. We are now distributing data from various Japanese broadband seismic networks (including PACIFIC21) together with data from BATS (Broadband Array in Taiwan for Seismology) and IRIS through the system.

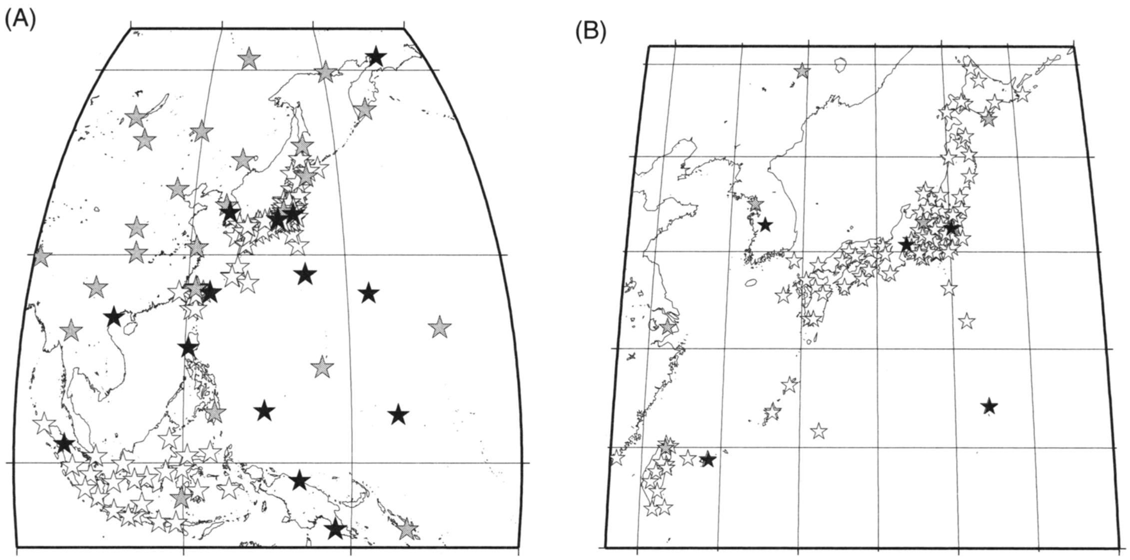
Digital seismic waveform data are being accumulated at an accelerating rate by various networks. Data analysis using these huge data sets is becoming increasingly common. For example, in the Western Pacific region, a large number of digital broadband seismographic stations have recently been installed by various institutions (Figure 1). If the data from these networks are appropriately combined, they effectively constitute a data set from a very dense broadband network. Such a data set should be very useful for determining seismic structure and earthquake source properties with high resolution. However, in the current waveform distribution system, such studies require somewhat tedious procedures for collecting the waveform data from different data centers, each of which has a different user interface.

Looking at the field of computer science, distributed object technology (technology for distributing software modules to multiple computers on a network; The Object Management Group, 2000) is one of the most important technologies in modern system-programming practice, as it can handle a huge system in a straightforward manner. Java RMI (Gosling and Steele, 1996; Sun Microsystems, 1999) provides one of the most successful technologies to build a distributed object management system because of its high capability for expansion and security.

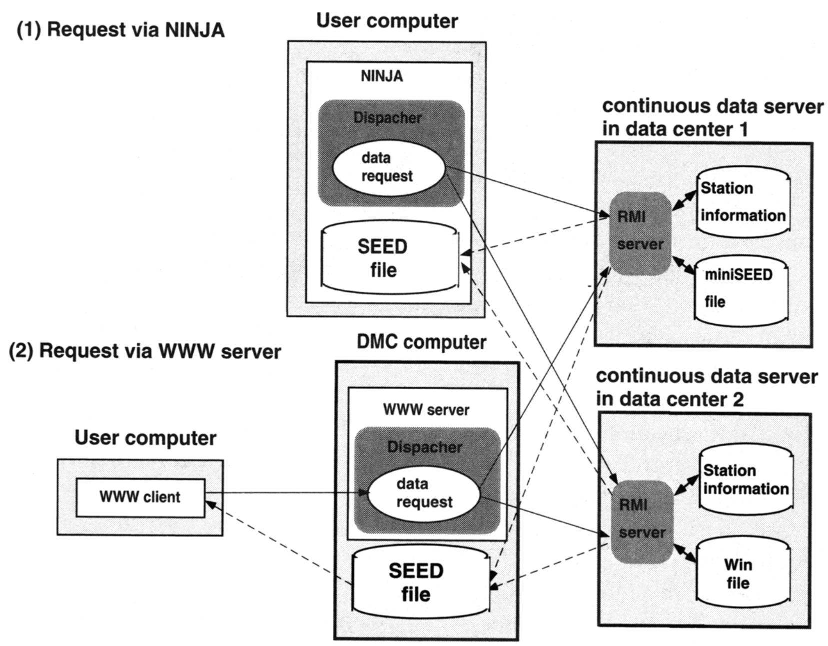
In this paper, we describe an application of Java RMI technology to develop a networked data distribution system that provides a unified interface for data from various networks. We show that our system is user-friendly and has highly capability for expansion. In the main part of this paper we present only the highlights of our system. The detailed specifications are given in the appendix. For simplicity, we discuss only the system for continuous data in the main part and refer the reader to the appendix for the structure of the system for event data.

OVERALL STRUCTURE

The overall system structure is shown in Figure 2, where the gray squares represent objects. In this system, seismic waveform data are managed by respective data centers so that they are always up-to-date. Various request interfaces and data servers are networked via the Internet. If a user sends a request to a data request interface ("dispatcher", the two left gray squares in Figure 2), the dispatcher dispatches the request to objects in various data centers ("RMI server", the two right gray squares in Figure 2), collects the requested waveform data in an internationally recognized standard for-



▲ **Figure 1.** Broadband seismographic station map (A) in the western Pacific region and (B) in Japan and Taiwan. Black stars indicate the OHP network, gray stars the IRIS network (IRIS/USGS, IRIS/IDA, and IRIS/CDSN), and white stars other networks.



▲ **Figure 2.** Block diagram of our networked data distribution system. Gray squares indicate objects. "Win file" is a data file in Japanese domestic format.

mat (SEED format; Federation of Digital Seismographic Networks *et al.*, 1993), and returns the SEED volumes to the user. Thus, a user can download all of the latest seismic waveform data with a unified interface.

The internal structure of the RMI server is as follows. The data managed by each data center include waveform data and station information. Station information is included in the SEED header: the latitude and longitude of each station, instrumental responses, and so on. Each data center can archive waveform data and station information in its own format because the RMI server at each data center is designed corresponding to the local format used. Once an RMI server receives a request from a dispatcher, the RMI server generates a waveform data file in SEED format and sends the SEED file back to the dispatcher. When the data format is a miniSEED file and station information is managed by the standard portable data collection center toolkit, "pdcc_toolkit" (see http://www.iris.washington.edu/manuals/pdcc_exp.html for details), we utilize the standard SEED volume producing software, "pod" (see http://www.iris.washington.edu/manuals/POD.html for details).

For the dispatcher we developed two types of request interface. One is an interface via special application software installed in each user's computer (the upper-left gray square in Figure 2). It is Java application software named *NINJA* (New Interface for Networked Java Applications). The other is an interface via a World Wide Web (WWW) server (the lower-left gray square in Figure 2). It is a Java applet running in a WWW browser. The functions of the Java application and applet are similar. The main difference between them is the place where the programs are stored: The application is stored in a user's computer, whereas the applet is stored in a WWW server and is downloaded into a user's WWW browser when that user accesses the server. Each dispatcher should be informed of the currently available RMI servers and data availability in each RMI server. All of this information should also be synchronized among dispatchers. We adopt the Directory Service to build the synchronization mechanism. The details are discussed in the section "Information Synchronization."

COMMUNICATION THROUGH FIREWALLS

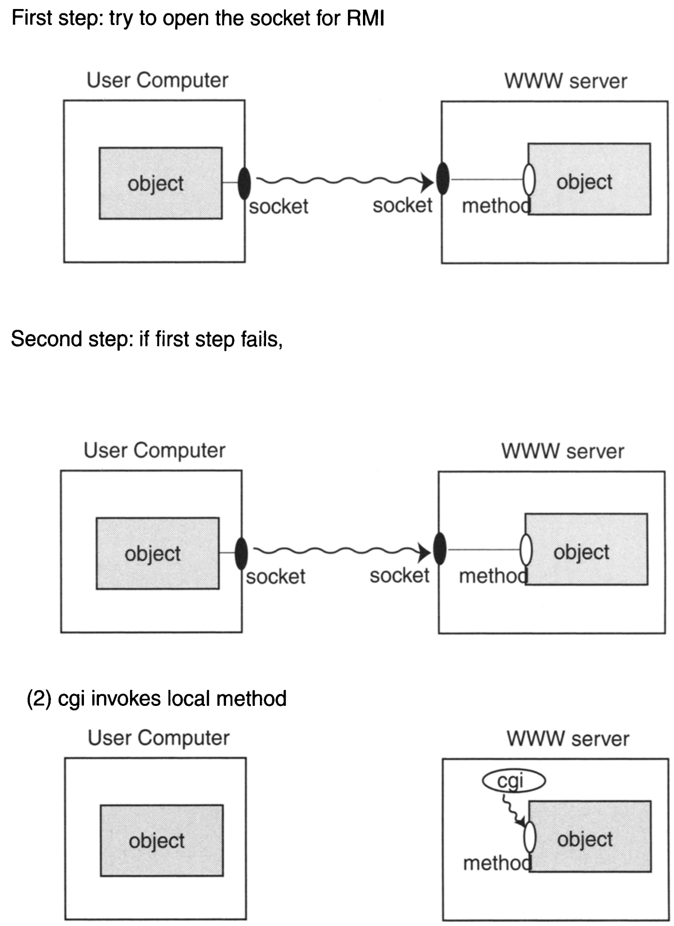
Because firewalls are increasingly common in Internet environments, communication through firewalls must be possible in a networked system. Java RMI has a default tunneling mechanism for firewalls. When Java RMI notices the existence of a firewall and finds it impossible to open the socket for RMI, it tries to reconnect using http (Sun Microsystems, 1999).

However, in our experience, the above mechanism does not work in many cases. The tunneling procedures are invoked when an ICMP (Internet Control Message Protocol) packet with the message "destination unreachable" is detected, but this packet often is not returned because of the specifications of the firewalls. In those cases, Java RMI incorrectly concludes that a route cannot be established and produces network error messages.

To solve this problem for communications between dispatchers and RMI servers, our dispatcher software sets a compulsory Java tunneling mode enable regardless of whether it receives an ICMP packet for "destination unreachable." However, for communication between a user's WWW browser and a WWW server, to set up this mode, users must properly set the properties of their own WWW browser by themselves. Also, for some browsers, such a change of properties causes a security-related warning message to be issued. This is not user-friendly, so we developed our own tunneling mechanism.

We adopt the filtering technique using cgi shown in Figure 3. First, a Java object in a user's computer tries to open an RMI socket on a remote WWW server. If that fails, it tries the following two-step procedure: First it invokes the cgi in the WWW server using an http connection, then the cgi invokes a local method.

As long as a user's browser can access the WWW server, the above tunneling mechanism should work because we use only standard http connections. Because users do not have to set up properties of the browser, this mechanism is user-friendly. If the standard Java tunneling mechanism works well in the future, we will dispense with our tunneling mechanism by removing the second step in Figure 3 from the source files. Thus the impact on the software should be small.

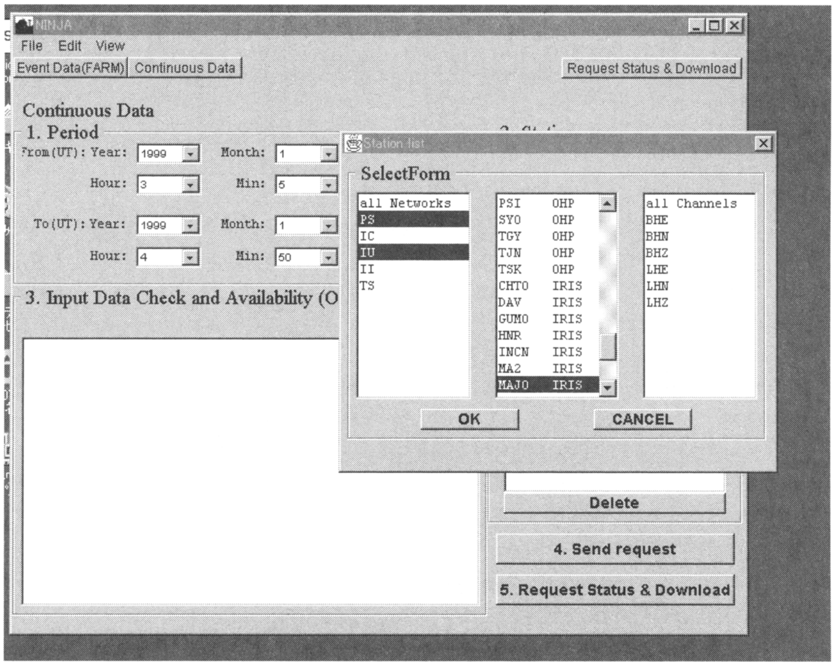


▲ **Figure 3.** Our firewall tunneling mechanism using cgi.

INFORMATION SYNCHRONIZATION

To optimize the load distribution among dispatchers, it is desirable to be able to set up an arbitrary number of dispatchers. If we have only one dispatcher, it suffices to have a local database for the information required in dispatching a request. The required information in this case consists of the URL's of RMI servers and data availability based on station and channel names in each RMI server. However, if we have multiple dispatchers, the information synchronization between the databases (*i.e.*, dispatchers) becomes a problem. One might think that one of the most straightforward methods is direct communication between dispatchers. However, this is almost impossible because we cannot know the URL of every dispatcher. (Note that application dispatchers, *NINJA*, are installed in each user's computer.)

In our system, each dispatcher refers to a database on the Internet using the Directory Service, an electronic database that contains information on entities and that is accessible over the network (see "Directory Service" in the Appendix for details). Thus all information is shared by every dispatcher.



▲ **Figure 4.** Data selection window in our application dispatcher (*NINJA*).

We build the mechanism such that RMI servers automatically and periodically report the updated information to the Directory Server. Thus all information is updated in semireal time. The Directory Server is set up independently from dispatchers and RMI servers. We adopt LDAP (Lightweight Directory Access Protocol; Yeong *et al.*, 1995) as the protocol to lessen the communication load.

Our system achieves a user-friendly request interface (Figure 4). Users do not have to set up anything for the URL of RMI servers and so on, because all information is automatically obtained through the network; the list of available data (network, station, and channel names) is compiled automatically. Because the information in the database is automatically updated (except for URL's of RMI servers), whenever a new RMI server is ready, new available networks, stations, and channels are automatically added without any special set-up.

This system also has advantages for system managers. The automatic update mechanism lessens the load for daily management. Setting up a new RMI server is straightforward because the information sharing system of the Directory Service does not require changing any settings of other dispatchers or RMI servers.

CONCLUSIONS

We have developed a networked data distribution system using Java RMI. The current available dispatchers and RMI servers are shown in Table 1. Two applet dispatchers and six RMI servers are now running. We have distributed many CD-ROM's on which the application dispatcher software is saved. They are available upon request to the first author of this paper (takeuchi@eri.u-tokyo.ac.jp). RMI servers exist for various Japanese broadband networks, BATS, and IRIS data. Available Japanese broadband networks are OHP (a part of PACIFIC21, Fukao *et al.*, 2001; formerly POSEIDON, Tsuboi, 1995), JMA, EOC, and GEOTOC data. See Appendix for an explanation of the network codes. In the future, we plan to distribute all PACIFIC21 data. Providing a unified interface for various network data will greatly improve the convenience for users. As our communication method and information synchronization mechanism does not require special set-up by a user, our system is quite user-friendly.

Our system also has high expansion capability. Java RMI is suitable for system expansion. A new function is easily developed and shared because it is architecture-neutral and we do not have to prepare source codes for each platform (Sun Microsystems, 1999). Because of our information synchronization mechanism our system is readily expanded. New dispatchers and RMI servers are easily added to our system because we do not have to change any setting of preexisting dispatchers or RMI servers. Our system can be applied not only to seismological data but to any geophysical time series data including geomagnetic, GPS, and superconducting gravimeter data.

ACKNOWLEDGMENTS

We thank Honn Kao and Wen-Tzong Liang for their installation of our software in Taiwan. We thank Tim Ahern for allowing us to install our system in the IRIS DMC. We also thank Yumiko Tanabe for setting up the database for various RMI servers. This research was partly supported by a grant from the Japanese Ministry of Education, Culture, Sports,

TABLE 1
Dispatchers and RMI servers (as of July 2001)

(1) Applet Dispatcher		
URL	**Location**	
http://ohpdmc.eri.u-tokyo.ac.jp/	Earthquake Res. Inst., Univ. Tokyo, Japan	
http://odin.earth.sinica.edu.tw/	Inst. of Earth Sci., Academia Sinica, Taiwan	
(2) Application Dispatcher (NINJA)		
We have distributed many copies of the NINJA CD-ROM.		
RMI Server (See http://ohpdmc.eri.u-tokyo.ac.jp/ninja/station.html for the current available station list.)		
Network	**URL**	**Description**
PACIFIC21/OHP	ohpsrv.eri.u-tokyo.ac.jp	Broadband network by Japanese Ocean Hemisphere Project
JMA	ocean.eri.u-tokyo.ac.jp	Broadband network by Japanese Meteorological Agency
EOC	ohpsrv.eri.u-tokyo.ac.jp	Broadband network by ERI (Earthquake Res. Inst., Univ. Tokyo)
GEOTOC	ocean.eri.u-tokyo.ac.jp	Ocean-bottom cable by ERI and IRIS
BATS	aeolus.earth.sinica.edu.tw	Broadband Array in Taiwan for Seismology
IRIS/USGS, IRIS/IDA, IRIS/CDSN	dmc.iris.washington.edu	Worldwide broadband network by IRIS and cooperating institutions

Science and Technology (the Ocean Hemisphere Project and No. 12740257). This research was also partly supported by the Japanese Earth Simulator Project.

REFERENCES

Federation of Digital Seismographic Networks, Incorporated Research Institutions for Seismology, and United States Geological Survey (1993). *Standard for the Exchange of Earthquake Data Reference Manual*, 203 pp.

Fukao, Y., Y. Morita, M. Shiobara, T. Kanazawa, H. Utada, H. Toh, T. Kato, T. Sato, H. Shiobara, N. Seama, H. Fujimoto, and N. Takeuchi (2001). The Ocean Hemisphere Network Project (OHP), in R. Romanowicz, K. Suyehiro and H. Kawakatsu (editors), *Workshop Report of OHP/ION Joint Symposium on "Long-Term Observations in the Oceans"*, Japan, 13–29.

Gosling, J., B. Joy, and G. Steele (1996). *The Java Language Specification*, Addison-Wesley.

The Object Management Group (2000). *The Common Object Request Broker: Architecture and Specification*, Revision 2.4.

Sun Microsystems (1999). *Java Remote Method Invocation Specification*, Revision 1.7.

Tsuboi, S. (1995). POSEIDON, *IRIS Newsletter* **16**, 8–9.

W. Yeong, T. Howes, and S. Kille (1995). *Lightweight Directory Access Protocol* (Request for Comments: 1777).

Earthquake Research Institute
University of Tokyo
Yayoi Bunkyo-ku
Tokyo 113-0032
JAPAN
takeuchi@eri.u-tokyo.ac.jp
(N.T., S.W., Y.F.)

Faculty of Science
Yokohama City University
Seto Kanazawa-ku
Yokohama 236-0027
JAPAN
(S.T.)

Research & Development Systems Department
Science Systems Division
Fujitsu Limited
Izumi-cho Mito 310-0026
JAPAN
(M.K.)

Research & Development Systems Department
Science Systems Division
Fujitsu Limited
Takezono Tsukuba 305-0032
JAPAN
(Y.M.)

Earth Science Systems Department
Science Systems Division
Fujitsu Limited
Nakase Mihama-ku Chiba 261-8588
JAPAN
(T.N.)

APPENDIX

SPECIFICATION OF THE SYSTEM

Structure for the System for Event Data

The geometry between dispatchers and RMI servers is the same as the geometry in the system for continuous data, which is shown in Figure 2. The data managed by each event RMI server are event FARM data (pre-assembled waveform data for significant earthquakes) and the event list archived at the RMI server. Event FARM data can be stored only in SEED format. Station information is not managed by event RMI servers.

First a user chooses a network and a year (or a month) to obtain a list of earthquakes from an event RMI server. More specific event selection is optional. A dispatcher sends the list request to the corresponding RMI server and receives the list of earthquakes of the year stored in the RMI server. The user then selects earthquake(s) from the list and the dispatcher forwards the data request to the RMI server. The dispatcher relays the event SEED volume(s) returned from the RMI server to the user.

The mechanisms for communication through firewalls and for information synchronization between dispatchers are the same as the mechanisms in the system for continuous data.

Available Data

OHP FARM and IRIS FARM are available (as of July 2001). We plan to create JMA FARM and EOC FARM. OHP FARM is made according to the same criteria as IRIS FARM (Table 2). Available continuous data are the data from various Japanese networks (OHP, JMA, EOC, and TPC-1) and BATS, IRIS/IDA, IRIS/USGS, and IRIS/CDSN (as of July 2001).

OHP is the network funded by the Japanese Ocean Hemisphere Project (see http://eri-ndc.eri.u-tokyo.ac.jp/ohp/ for details) and operated by universities and other cooperating institutes in Japan. OHP is part of PACIFIC21 (a successor of POSEIDON), which is a generic name for a database of various Japanese broadband seismographic data (see http://pacific21.eri.u-tokyo.ac.jp for details). The number of OHP stations is about 15, and they cover mainly the western Pacific region. Most of the sensors used are STS-1.

JMA is the network operated by the Japanese Meteorological Agency. The network is mainly for routine real-time monitoring of relatively large earthquakes. The number of stations is about 20, and they uniformly cover all Japan. The sensors used are STS-2.

EOC is the network operated by the Earthquake Observation Center, which is one of the divisions of the Earthquake Research Institute at the University of Tokyo. It is a network funded by the new Program of the Study and Observation for Earthquake Prediction. The number of stations is about 20, and they cover mainly the central Japan area (Kanto-Koshinetsu area). Most of the sensors used are CMG-3.

GEOTOC is an ocean-bottom cable installed through the cooperation of ERI and IRIS. The cable connects the city of Ninomia, located in the central part of Japan, and Guam. The sensor is a type of accerelometer used for aviation navigation.

Request Interface

The above data are distributed upon requests from users automatically through the applet or application dispatchers. An e-mail interface (request1@ohpdmc.eri.u-tokyo.ac.jp) is also available for continuous data requests.

The user chooses either direct download, anonymous FTP, or tape as a download method by clicking the "DOWNLOAD", "FTP", or "TAPE" button. If anonymous FTP is selected, the user will be notified by e-mail from the data center that the data set is available on our anonymous FTP site. If tape is selected, a tape containing the requested data will be mailed from the Ocean Hemisphere Data Management Center.

E-mail requests should be written in the IRIS BREQ_FAST format (see http://www.iris.washington.edu/manuals/breq_fast.html for details) and requested data will be downloaded through anonymous FTP. The format of all outgoing seismological data is SEED.

Other Useful Functions

In the event data window, users can search the events satisfying specified criteria by clicking the "Options" button. Time window, earthquake depth, size (moment magnitude), location, region, and distance from a particular station can be specified as selection criteria. The region is specified by clicking the block in the map which appears in a pop-up window. The distance from a particular station option allows a user to search for events whose epicenters are within a specified distance from the specified station. A user can reset the criteria by clicking the "Reset" button. The event list satisfying the criteria will be displayed by clicking the "Search" button.

In the continuous data window, a user can view all continuous waveform data before downloading by clicking the "view waveform" button (for applet version dispatcher only). One can also check the consistency of input parameters (*e.g.*, checking if starting time is not prior to the ending time) and the availability of waveform data by clicking the "Check" button.

Directory Service

Although we can set up an arbitrary number of Directory Servers, we currently have only one Directory Server located in the Earthquake Research Institute at the University of Tokyo. The entries stored in the Directory Server are URL's of RMI servers and data availability based on station and channel name in each RMI server. URL's of RMI servers are registered in the Directory Service by hand. Data availability is automatically reported from each RMI server every 24 hrs after it is booted. A dispatcher refers to the Directory Server whenever the dispatcher compiles the available data (*i.e.*, when it displays the available network names in the event data request interface and when it displays the available network, station, and channel names in the continuous data request interface).

TABLE 2
Criteria for Making OHP FARM

M_w	SEED Channel	Starting Time	Ending Time
($5.5 \le M_w \le 5.7$ and Depth ≥ 100 km) or $5.8 \le M_w \le 5.9$	BH	120 s before first P	3600 s after first P
	LH	600 s before first P	600 s after R_1 arrival whose apparent velocity is 3 km/s
$6.0 \le M_w \le 6.9$	BH	120 s before first P	3600 s after first P
	LH	600 s before first P	600 s after R_3 arrival whose apparent velocity is 3 km/s
$7.0 \le M_w \le 7.9$	BH	120 s before first P	3600 s after first P
	LH	600 s before first P	600 s after R_5 arrival whose apparent velocity is 3 km/s
$8.0 \le M_w$	BH	120 s before first P	3600 s after first P
	LH	600 s before first P	600 s after R_7 arrival whose apparent velocity is 3 km/s