A Web-based on-line Monitoring and Diagnosis System for Machinery of Hot Strip Mill

Hung-Chieh Yeh^, Chung-Nan Lee^, Song-Jau Tsai*, Chuan-Wen Chiang^, Ming-Shen Jen^, Chang-Tsun Lin*, Cheng-Yu Wei*, Shaw-Ching Chang#, Der-Lin Wang#, Kun-Hong Shieh#, and Sheng-Yang Lin*

^Department of Computer Science and Engineering, National Sun Yat-Sen University,
*Steel and Aluminum Research and Development Department, China Steel Corporation,
#South Area Information Division of Institute for Information Industry
Kaohsiung, Taiwan

Abstract

A multi-agent based on-line monitoring and diagnosis system (MAMDS) is developed for distributed measuring, monitoring and diagnosis machines of a hot strip mill in the China Steel Corporation. MAMDS takes the advantage of Web-based properties, it releases the restrictions of temporal and regional isolation and provides a platform-independent user interface. Through the system, users can easily monitor and diagnose the machines on remote production lines through its graphical user interfaces, such as display of machine status, malfunction alarming, trend chart, waveform, and spectrum. It uses collaborative agents to reduce the cost of development and enhance the feature of reuse. The system provides a real-time and dynamical operation circumstance and can meet the needs of different levels of users. Consequently, it is of great help to run machines smoothly, lower the cost of personnel and enhance the ability of market competition.

Keywords – monitoring and diagnosis, web-based system, multi-agents, plant automation

1. Introduction

As modern industrial facilities move toward high speed and automation, the on-line monitoring and diagnosis become important. Since a production line facility may break down unexpectedly, the unscheduled shutdown will cause losses and deteriorate the quality ofproducts. The traditional time-based maintenance strategy does not take the practical machine condition into account and only performs the maintenance practice at a fixed time interval from the maintenance personnel. Therefore, accidental breakdowns unavoidably happen, when the chosen time interval is too long; otherwise, it leads to over-maintenance, when the chosen time interval is too short [1].

In order to increase industrial automation, industrial monitoring and diagnosis system based on computer technology gains popularity [2-4]. In general, monitoring the machines or supervision is carried out by measuring some data from some machines in the plant production lines. The development of on-line monitoring and diagnosis system is prompted by the needs to increase the availability of machines and to protect machines from problems. Furthermore, maintenance personnels can access the information that is provided from a machine monitoring system to check whether these machines are running normally.

As the Internet is in widespread use, many enterprises take advantage of the Internet to extend services without temporal and regional restrictions[5,6]. Therefore, the trend has brought about changes in the manufacturing procedure. The distributed measurement system is gradually developed for carrying out remote control and information retrieval on machines to be monitored. In general, those systems are built as traditional client/server architecture. But such architecture has some restrictions on the use and development of the system, for example, a user has to often install updated software on the operating platform often. Consequently, when the number of clients grows gradually, the management of versions and compatibility of application softwares on different operating platforms become tedious.

In recent years, system developers of large applications adopt agent technology; it simplifies functions of the system by the cooperation of agents [3,7-9]. Those agents in their executing environment can work asynchronously and autonomously [10]. Furthermore, they can work with each other for an objective requested by a user. In this paper we use a collaborative software agent to achieve the objective of MAMDS. By collaborative agents, the modification of MAMDS becomes simple and easy. According to the objectives of the users' requests, some designated agents will be able to handle the requested.

In order to release the restriction of temporal and regional isolation, MAMDS is developed as a Web-based system too. Hence, some problems such as exchanging information, sending/receiving messages, and uploading/downing files through a standard web browser need to be taken care of. Furthermore, the system has been developed over the years. It needs to exchange and interact information among agents implemented in different programming languages.

The remainder of the paper is organized as follows. Section 2 presents the architecture of MAMDS. Section 3 describes the analysis and design of MAMDS. The issues of implementation are described in Section 4. Demonstrations illustrating the operating production line of a hot strip mill are given in Section 5. Section 6 concludes the paper.

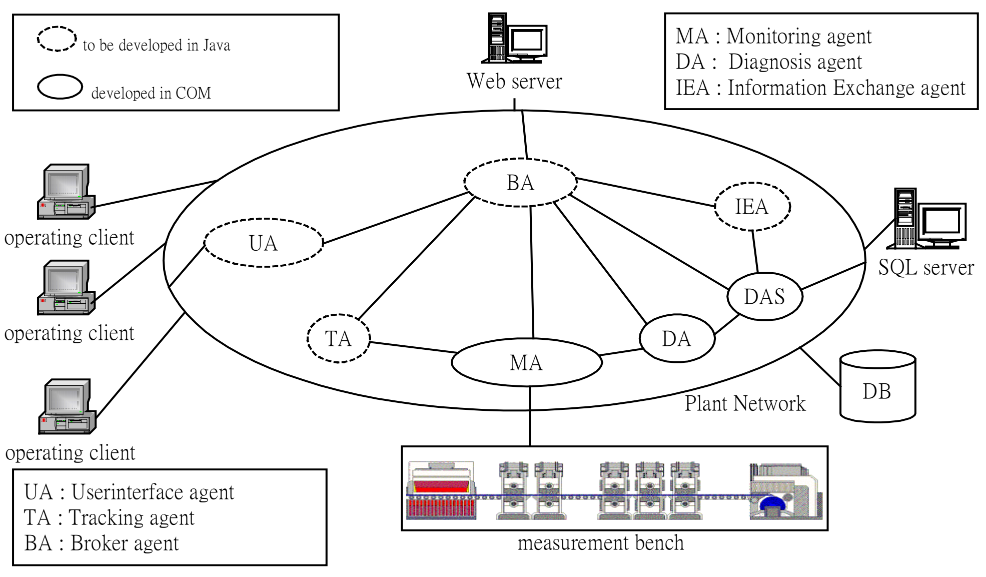


Fig.1 The system architecture of MAMDS

2. The Architecture of MAMDS

In order to fulfill users' requests and requirements, MAMDS must complete each request and requirement by the autonomous, cooperative, and coordinated agents. Agents in MAMDS can be implemented in different programming languages, for example, Java or COM (Component Object Model). As illustrated in Fig.1, the MAMDS architecture consists of agents, web server, database server, and measurement bench for operating machines of a production line. The agents in MAMDS can be divided into three tiers: the application tier, the middle tier, and the data tier as shown in Fig. 2. In the following paragraphs, the tasks and functions of each agent will be first described, followed by a discussion on the relationships of agents.

- Monitoring agent (MA): It is capable of performing measurement on monitored machines of a production line in an independent way or accepting a measurement command at any time. It performs measurement independently according to different measured points at different measurement modes. Those measurement modes are either at constant time interval or at some triggered condition. The measurement mode can be configured by a user.
- Diagnosis agent (DA): It is responsible for accepting the measurement data from MA. And then, it compares the up-to-date measurement data with some defined threshold values and operations to generate diagnosis results that consists of statuses, and preliminary diagnosis messages. The defined threshold values can be modified and stored into a database.
- Data acquirement and storage agent (DAS): This agent is employed to store information into a database. The information consists of raw measurement data provided from MA, post-processing analysis data provided by DA, and setting data from each user.
- Broker agent (BA): This agent acts as a broker for all other agents at different tiers. The main responsibility of this agent is to send requests and receive process results between the application tier and the data tier. In addition, this agent coordinates all requests to solve the synchronous accessing problem.
- User interface agent (UA): In order to interact with users smoothly, this agent provides sequences of manipulation graphic user interfaces. This agent constructs user interfaces in accordance with the model-view-controller (MVC) pattern to help to display and to manage visualization information. Besides, it is responsible for sending a user's requests to servers.
- Tracking agent (TA): In order to respond to the status and measurement information of monitored points at any time, this agent must update the up-to-date information continuously. At the application tier, it must track communication between a client and a server and notify UI agent when visualization information is updated. Besides, in order to send the up-to-date information to current on-line clients, this agent also tracks communication conditions of current clients.
- Information Exchange agent (IEA): This agent is responsible for wrapping heterogeneous information into Java-based agents. Therefore, agents implemented in Java can retrieve the same information provided by other agents implemented in COM. Besides, this agent must ensure message passing correctly and smoothly among agents implemented in Java and COM, so that the measurement data can be received and updated.

Moreover, MAMDS operates and provides monitoring and diagnosis services at a plant network. It consists of database server, measurement bench, and a Web server. Database is responsible for storing measurement and diagnosis data and providing historical measurement data for pre-diagnosis. Besides, Web server can provide transmission of web pages and applets to achieve files uploading/downloading and execute applets. Measurement bench provides measurement data during runtime of machines of a production line.

Figure 2 shows the relationships of agents in MAMDS during user operating runtime. During runtime the monitoring agent, the diagnosis agent, the data acquirement and the storage agent, the information exchange agent, and the tracking agent are operating in the data tier to collect measurement information and wait for users' requests. The broker agent is running at the middle tier to act as an information exchange broker. The user interface agent and the tracking agent operate at the application tier to allow a user to manipulate monitoring information collected from machines of a production line.

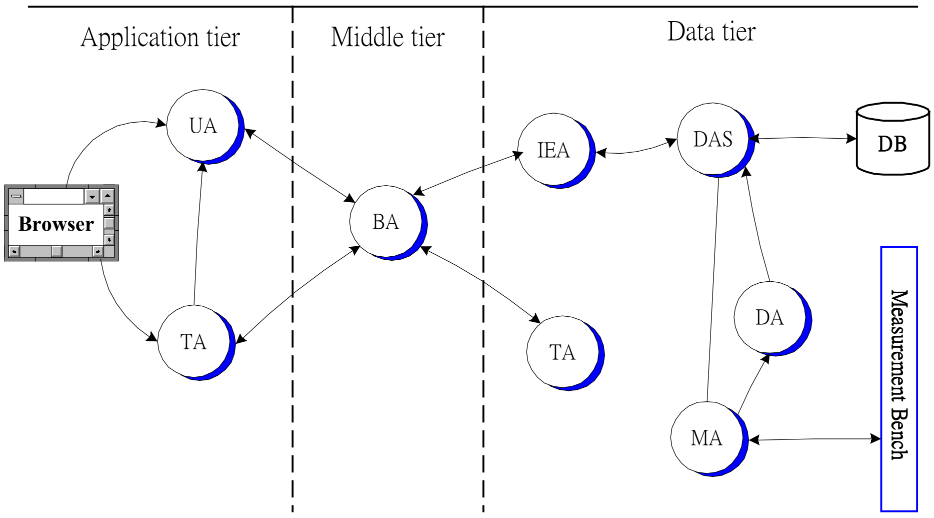


Fig.2 The relationships of agents in MAMDS

2.2 Measurement Bench

The core of MAMDS is a measurement bench that takes all measurement information corresponding to all measurement points on the machines of a production line. The measurement bench connects to all monitored machines and collects measurement information continuously during the operating time of a production line. MAMDS measurement bench of a hot strip mill in the China Steel Corporation (CSC) uses the Distributed Input/Output Controllers (DIOCs)[1] which are data collection devices designed by CSC. MA of MAMDS is made up of the DIOCs. MA measures the signals coming from various sensors and transmits the digitized signals through the plant network to DA and DAS. The measurement bench consisting of a number of DIOCs and hardware units is shown in Fig.3.

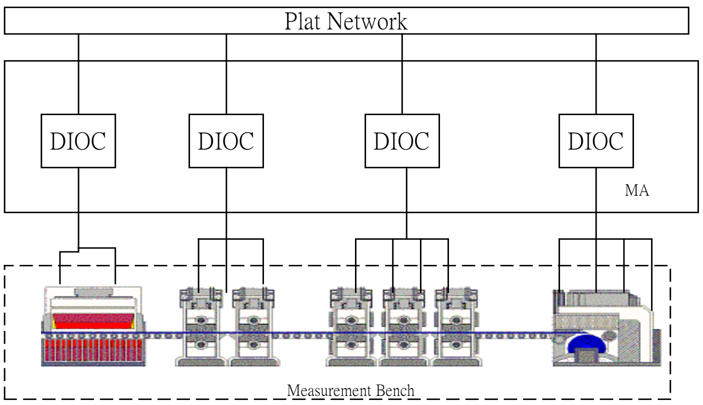


Fig.3 The measurement bench for

DIOC is a data collection device that is installed in the vicinity of the machines to be monitored. Its main functions include:

(1) Providing various kinds of signal process modules for preprocessing of the measured signals
(2) Diagnosing of the signal property to judge whether the signals are short-circuited or open.
(3) Accepting commands to execute measurement.
(4) Digitizing measured signals.

3. Analysis and Design

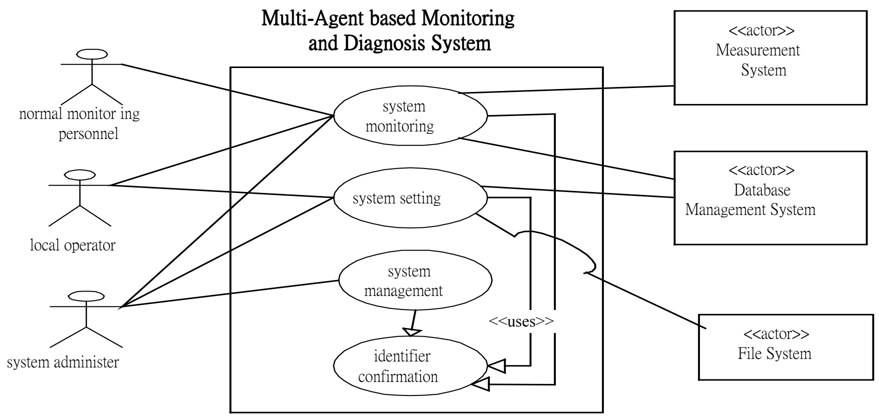
The Unified Modeling Language (UML), which is proposed by Rational Software Cooperation, Grady Booch, James Rumbaugh, and Ivar Jacobson [11] is widely accepted for the representation of developing enterprises or large-locale?? software systems. It allows users to designate, represent, construct, and record various components in systematization procedures.

We follow the analysis and construction concepts of UML to analyze and design the on-line monitoring and diagnosis system. The use case model in UML is used to represent scenarios of system's functions.

3.1 The Use Case Model

The use case in UML specifies actions for a system to interact with external actors including users and the system. It describes the main functions including the system, high-level system architecture, external actors, and then interaction relationships. The use case model for MAMDS is shown in Fig.4.

The levels of authority of the system for external actors (users) are classified into normal monitoring personnel, local operator, and system administrator. MAMDS offers functions, such as system monitoring,



Multi-Agent based Monitoring and Diagnosis System
normal monitor ing personnel
local operator
system administer
system monitoring
system setting
system management
identifier confirmation
<<uses>>
<<actor>> Measurement System
<<actor>> Database Management System
<<actor>> File System

Fig.4 The use case model for MAMDS

system setting, system management, and identity confirmation. Identity confirmation is responsible for verifying the identification of each user and for determining the levels of authority of the user. Then, it decides what kinds of functions and data a user can access. System monitoring provides a real-time and current operating status. Since some functions are only for acquiring information, it can be used for all users. System setting includes measuring methods, measurement time interval, and companion basis of diagnosis about a measurement point. In addition, system setting provides modifying current monitored points, such as adding and removing. An administrator can edit the users' data and modify a user's level of authority through system management function. The data storage repository is used by database management function and file system function to store the information of the production line, users of the system, and other system data settings.

3.2 Operational Phase Analysis

MAMDS mainly uses operational phase to achieve monitoring and diagnosis objectives. Operational phase includes monitoring and diagnosis, login checking, data browsing, file uploading, file downloading, measurement mode configuration, and information updating. Functions in an operational phase are as follows:

(1) Monitoring and diagnosis: MA and DA are activated unless all machines stop operating. MA must continuously measure points to be monitored either at constant time interval mode or at triggered condition mode. In addition, MA also can take measurement on behalf of a user's request. When MA takes measurement data, it transmits measurement data to DA for diagnosis. MA transmits raw measurement data to DAS. And DA also sends the diagnosis results to DAS.

(2) Login checking: It determines a user's validity using the levels of authority. The levels of authority classify the levels of users, from which it protects the system from inappropriate operations by unauthorized users that might lead to system and security problems or unnecessary information provided by normal users. When a user completes login phase, UA must notify TA to record this UA information in the data tier via BA in order to display the UA updated information. MAMDS can transmit and notify current measurement information of measurement points via TA in the data tier.

(3) Data browsing: UA presents monitoring and diagnosis information in terms of waveform, spectrum, and trend chart. UA can receive the messages about visualization information that is updated from TA at the application tier. It helps to display the real-time measurement information to a user by the cooperation of UA and TA.

(4) File uploading and downloading: In order to download the collected data from measurement devices by operators, MAMDS permits uploading files or downloading files that contain measurement results. At the same time, operators can download files that contain the historical measurement data. This phase is completed by the cooperation of BA and DAS.

(5) Measurement mode configuration: In order to retrieve measurement data from monitored points on machines of a production line, MAMDS allows to configure the measurement mode that consists of measurement time interval and condition. After UA receives configuration requests information from a user, it transmits the configuration requests information to DAS via BA.

(6) Information updating: MAMDS mainly executes monitoring and diagnosis. At the same time, TA at the data tier is notified what measurement point has measurement information to be updated, and whether MAMDS has on-line users. Finally, TA transmits the up-to-date information to the front-end via BA.

4. Issues of Implementation

This section describes the implementation of MAMDS for machines. MAMDS has been developed over several years. During the period of development, different languages have been used for different agents. For example, UA, TA, BA and IEA in MAMDS are implemented in Java language. MA, DAS, and DA are implemented in COM. Java language provides several characteristics, such as platform-independent as “write once, run anywhere”, powerful network ability and distributed computing, and integration. Besides, the class files of Java program can move through the Internet easily and run through any standard web browser.

Inside MAMDS it has to exchange and interact information among agents either implemented in COM or in Java. To achieve the requirements and objectives of the system, many implementation issues, such as information exchange, security, and data consistency must be solved. Details of each issue are discussed in the following subsections.

4.1 Information Exchange

In order to retrieve current and historical information of monitored machines from MA and DAS implemented in COM, IEA of MAMDS is responsible for handling information heterogeneity among agents. Since data type and message passing are different between Java and COM, it may cause some problems. IEA uses Java Native Interface (JNI) to implement information exchange function. Therefore, MAMDS provides a platform-independent monitoring and diagnosis graphical user’s interface by the characteristics of Java and function of IEA.

With information technologies that grow gradually, and protocol and programming languages become more diverse, APIs of Java alone may not be enough to satisfy a wide variety of applications. For this reason, Java provides JNI to communicate among other programming languages. In order to communicate among agents implemented in Java and COM, IEA must be responsible for information exchange and receiving all events for the updating measurement information of monitored machines.

IEA uses the techniques provided in [10,11] to handle different data type exchanges between Java and COM to communicate and collect measurement information of monitored machines among agents. Second, IEA must receive all events for updating measurement information of monitored machines validly to protect from measurement information loss. However, MA is implemented in COM and fires window’s events for updating information of monitored machines to a hide window. In other words, MA notifies updating information via a running background window. Therefore, IEA is responsible for communicating with this background window to build a communication bridge between MA and IEA. The steps for IEA to receive all events for updating measurement information from MA are as follows:

Step 1: At the initiation, IEA and MA must activate. MA is responsible for collecting the measurement information of monitored machines continuously. However, IEA builds a background window to receive all events.

Step 2: When MA completes a measurement task, MA fires an updating measurement event to this background window.

Step 3: After this background window receives an updating measurement event, this background window finds current running Java Virtual Machine (JVM) and retrieves JNI via JVM. Afterwards, this background window can send an updating measurement message to IEA.

Step 4: IEA receives an updating measurement message and then retrieves the updated measurement information by the updating measurement message.

4.2 Security

MAMDS must be accessed from a standard web browser. However, a standard web browser has to assume that an applet is not trustable in order to protect a host from downloading and executing an unsafe Java applet. It restrains the use of a web browser in the following ways [12].

✓Accessing file system through a web browser

✓Executing a native code

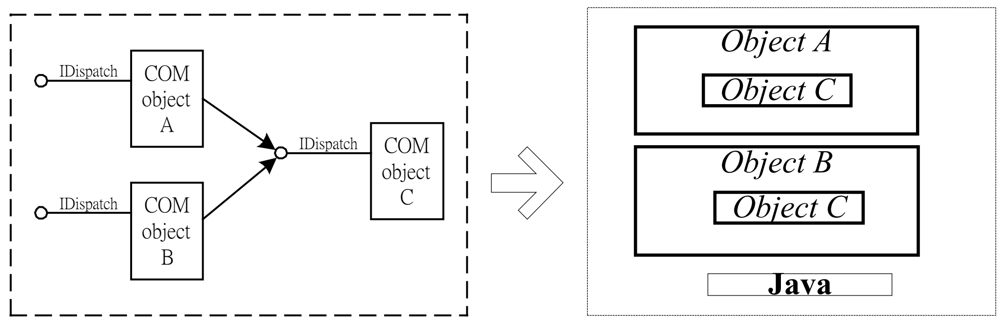


Fig.5 COM objects transform to Java objects

✓Reading certain system properties

✓Receiving incoming socket connections

Based on the restraints mentioned above, the use of general applet to satisfy our requirements is impossible. In order to satisfy the requirements of uploading or downloading files of measurement data, BA in MAMDS is implemented in the Java Servlet. Relying on Servlet can accomplish uploading/downloading measurement data in a binary file and reducing the cost of development.

4.3 Data Consistency

Measurement data are stored in a database and accessed via DAS implemented in COM, which are accessed through pointers. It is quite simple to maintain those common data through these pointers in COM. During exchanging and updating of data, the system only needs to update the up-to-date information once. However, Java is designed for simplicity in the initial stage, so it discards all pointer references. Therefore, it is hard to directly access data through pointers for agents implemented in Java. In order to maintain raw relations of data that is accessed through functions of pointer, we use Java objects to represent original data object in COM. These relations among common data objects accessed through pointers in COM must be handled.

We have to build the Java objects that correspond to COM's data models, and translate data from COM's data types to Java's data types by Java Native Interface (JNI). However, in order to avoid data duplication as illustrated in Fig.5, we must construct the relation of Java objects corresponding to the relation of objects in COM through pointers.

In this situation, MAMDS must update all objects which are related to this exchanged object for transforming COM objects to Java objects through JNI directly. Besides, data duplication may cause more usage of capacities. In order to solve the above

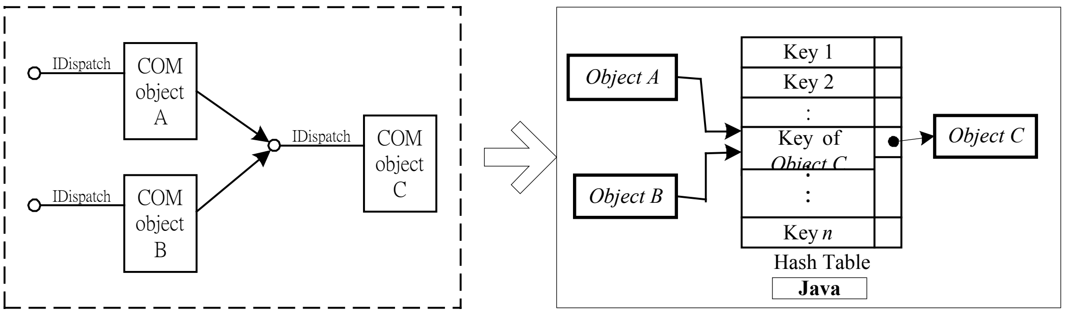
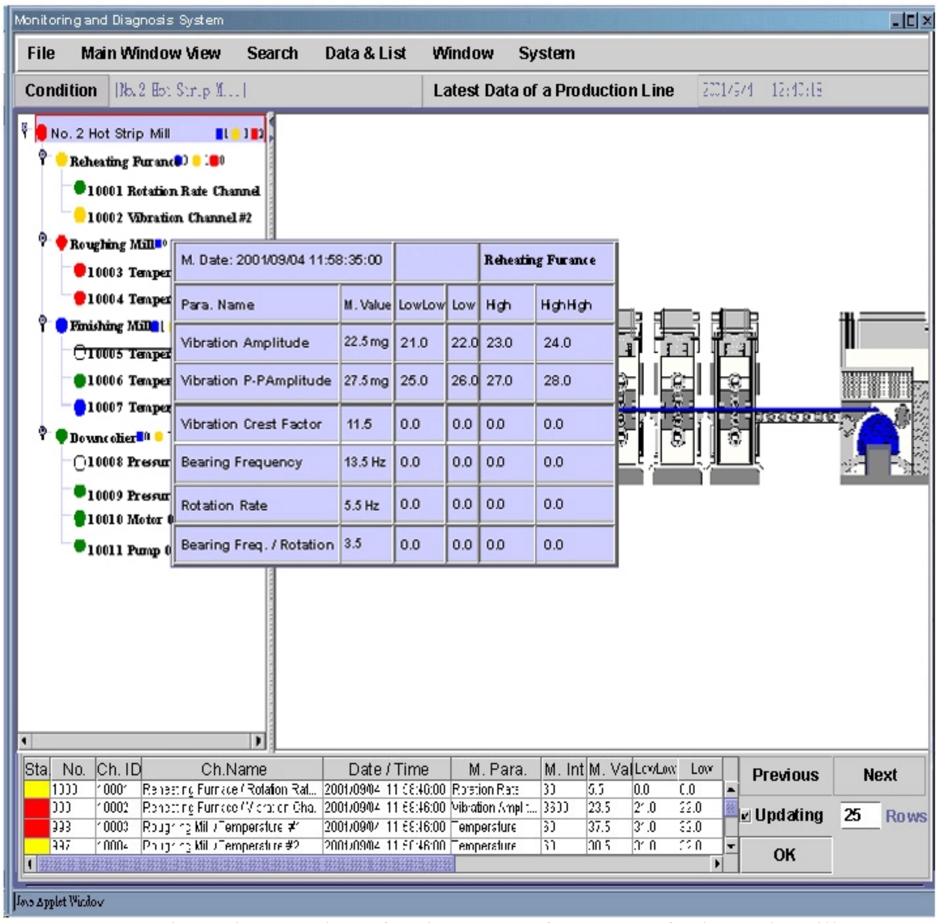


Fig.6 An improved data structure, which transforms a COM

problems and to keep the relation of objects through pointers in COM, we reduce the usage of capacity by rebuilding the relations of data and modifying the way of data exchange between Java and COM. We store every data as Java's object once and establish relational reference to a data hash table structure, which uses 'id's in COM data types as a key to retrieve the related object through the established hash table. An improved data structure of Fig. 5 is shown in Fig.6.

5. Demonstration of MAMDS

In this section, we demonstrate MAMDS of a hot strip mill currently used on the China Steel Corporation. Figure7 shows a main screen snapshot of MAMDS. The colored rectangle in the left top corner of the screen represents the current conditions of the machines present in the whole production line; green stands for normal, yellow for alarm and red for danger. In order to attract the user's attention, the colored



M. Date: 2001/09/04 11:58:35:00				Reheating Furance	
Para. Name	M. Value	LowLow	Low	High	HighHigh
Vibration Amplitude	22.5 mg	21.0	22.0	23.0	24.0
Vibration P-PAmplitude	27.5 mg	25.0	26.0	27.0	28.0
Vibration Crest Factor	11.5	0.0	0.0	0.0	0.0
Bearing Frequency	13.5 Hz	0.0	0.0	0.0	0.0
Rotation Rate	5.5 Hz	0.0	0.0	0.0	0.0
Bearing Freq. / Rotation	3.5	0.0	0.0	0.0	0.0

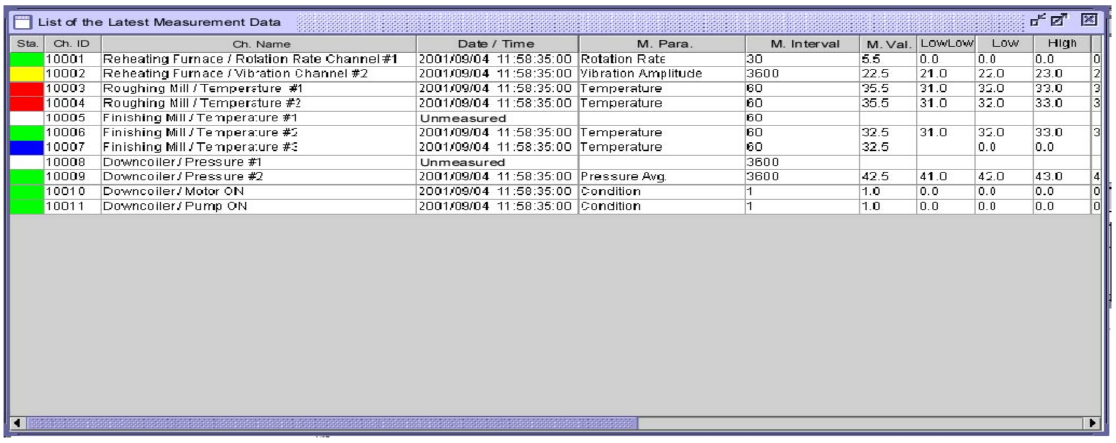
Sta.	No.	Ch. ID	Ch.Name	Date / Time	M. Para.	M. Int	M. Val	LowLow	Low
🟨	1000	10001	Reheating Furnace / Rotation Rat...	2001/09/04 11:58:40:00	Rotation Rate	30	5.5	0.0	0.0
🟥	[illegible]	10002	Reheating Furnace / Vibration Cha.	2001/09/04 11:58:46:00	Vibration Ampl...	3300	23.5	21.0	22.0
🟥	993	10003	Roughing Mill / Temperature #1	2001/09/04 11:58:16:00	Temperature	30	37.5	31.0	33.0
🟨	997	10004	Roughing Mill / Temperature #2	2001/09/04 11:50:46:00	Temperature	31	30.5	[illegible]	[illegible]

Fig.7 The snapshot of main screen of MAMDS for hot strip mill

rectangle keeps on blinking unless all unconfirmed alarms are checked. The hierarchical tree for this production line is displayed on the left-hand side of Fig.7. The hierarchical tree shows the workflow of the production line and each measurement points of facilities. Each hierarchical node contains one icon with different colors representing current status of measured points or production line. Therefore, it provides concise message for measurement data of this monitored and measured points. The popup table lists the measurement information for an arbitrary node, once it is pressed. On the right-hand side of Fig.7, a clear image of the production line is shown. When the hierarchy node representing a measurement point stops measuring, a solid line is displayed at the middle of a hierarchy node. At the right hand side, this system uses transparent buttons on a picture or an image to represent the workflow structure or facilities' locations of the production line. These pictures or images correspond to the tree node on the left hand side. In addition, it synchronizes with the tree on the left hand side.

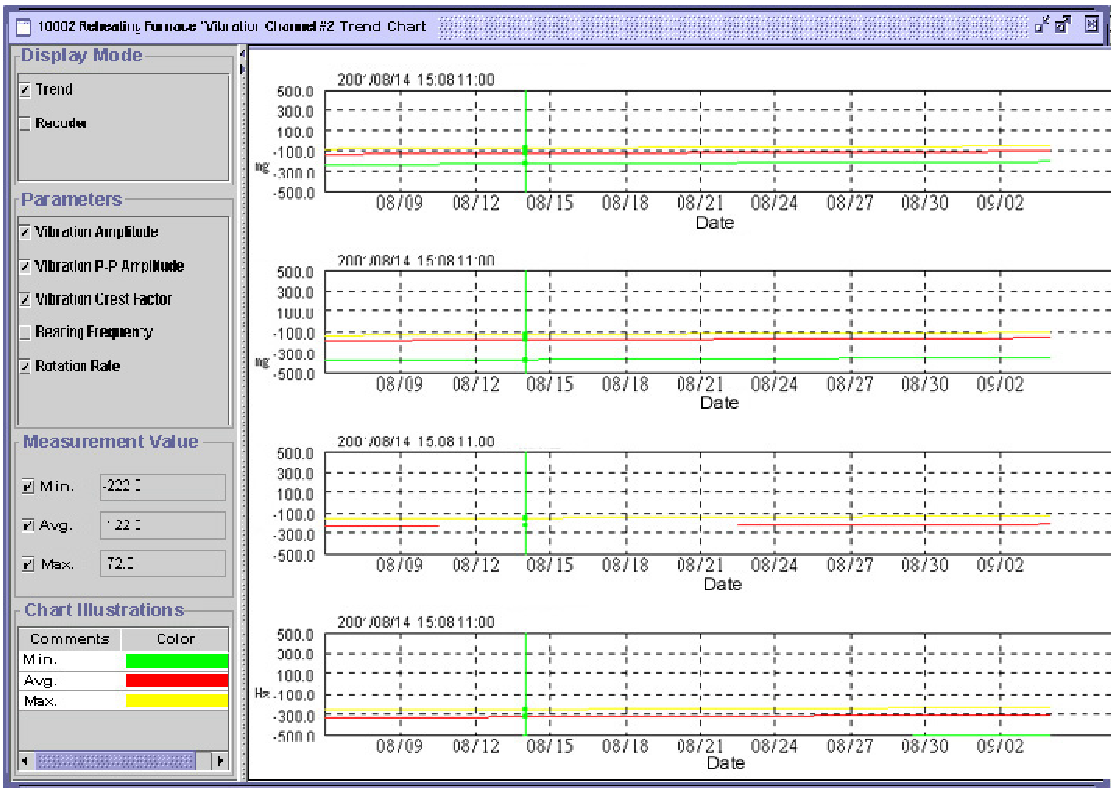
In order to understand the current status of each monitored and measured point, the system lists the information of all measurement points in a table, as shown in Fig.8. The information of each measurement point consists of name, the latest measured data (measured data and time, measured parameters, measurement value, etc.), preliminary diagnosis, and the types of measured data of the channel. The color in the status column of each row represents the condition of the latest measurement data. The "yellow" or "red" colors represent an alarm or a danger message, respectively. In order to trace alarm or danger measured signals for later use; these measured signals are stored in the database automatically. The other measured signals with normal machine condition can also be saved manually.

When the operating time of production line increases, the system can analyze the records of measurement data of each measurement point and can transform them into useful statistical data. These statistical data help users to observe the trend of each measurement point and the operating condition of facilities and to provide forecast information in accordance with this trend Information, which is represented into a chart as illustrated in Fig.9.


List of the Latest Measurement Data

Sta.	Ch. ID	Ch. Name	Date / Time	M. Para.	M. Interval	M. Val.	LowLow	Low	High	
🟩	10001	Reheating Furnace / Rotation Rate Channel #1	2001/09/04 11:58:35:00	Rotation Rate	30	5.5	0.0	0.0	0.0	0
🟨	10002	Reheating Furnace / Vibration Channel #2	2001/09/04 11:58:35:00	Vibration Amplitude	3600	22.5	21.0	22.0	23.0	2
🟥	10003	Roughing Mill / Temperature #1	2001/09/04 11:58:35:00	Temperature	60	35.5	31.0	32.0	33.0	3
🟥	10004	Roughing Mill / Temperature #2	2001/09/04 11:58:35:00	Temperature	60	35.5	31.0	32.0	33.0	3
⬜	10005	Finishing Mill / Temperature #1	Unmeasured		60					
🟩	10006	Finishing Mill / Temperature #2	2001/09/04 11:58:35:00	Temperature	60	32.5	31.0	32.0	33.0	3
🟦	10007	Finishing Mill / Temperature #3	2001/09/04 11:58:35:00	Temperature	60	32.5		0.0	0.0	
⬜	10008	Downcoiler / Pressure #1	Unmeasured		3600					
🟩	10009	Downcoiler / Pressure #2	2001/09/04 11:58:35:00	Pressure Avg.	3600	42.5	41.0	42.0	43.0	4
🟩	10010	Downcoiler / Motor ON	2001/09/04 11:58:35:00	Condition	1	1.0	0.0	0.0	0.0	0
🟩	10011	Downcoiler / Pump ON	2001/09/04 11:58:35:00	Condition	1	1.0	0.0	0.0	0.0	0

Fig.8 Display of the latest measurement data


10002 Reheating Furnace Vibration Channel #2 Trend Chart

Display Mode

- [✓] Trend
- [ ] Recorder

Parameters

- [✓] Vibration Amplitude
- [✓] Vibration P-P Amplitude
- [✓] Vibration Crest Factor
- [ ] Bearing Frequency
- [✓] Rotation Rate

Measurement Value

- [✓] Min. -222.[illegible]
- [✓] Avg. ·22.[illegible]
- [✓] Max. 72.[illegible]

Chart Illustrations

Comments	Color
Min.	🟩
Avg.	🟥
Max.	🟨

Fig.9 Trend chart of a measurement point

6. Conclusions

In this paper, we have constructed a Web-based cooperative multi-agent system -- MAMDS to enhance industrial automation. It may release restrictions such as geographical isolation, and operating time. It enhances flexibility for satisfying the changeable and multiform requirements and reducing the troubles of updating the application software in client sites. The system integrates software modules developed at different phases, as a result, it also enhances the reuse of industrial software and then reduces the cost of development by the collaborative agents. MAMDS offers the users a better operating environment for monitoring machines of a hot strip mill. In addition, the system assists users to perform maintenance and solve harmful problems of machines. This system can provide real-time information of monitored machines accurately and response to machine's condition during the operating period. By the historical data recorded at every measurement time, it provides users the operating trend of each machine and allows operators to take precautions to prevent machines from becoming deteriorated. As a result, it avoids machine damages and problems of product quality. Consequently, this system can assure smooth operation of the facilities, reduction of damage of accidental breakdown, and maintenance of product quality at the desired level.

Acknowledgements

This research was supported by South Area Information Division of the Institute for Information Industry and the Automation & Instrumentation System Development Section, Steel & Aluminum R&D Department of China Steel Corporation.

References

[1] S. J. Tsai, C. T. Lin, J. J. Jeng, C. Y. Wei, K. M. Chang, C. C. Chang, C. H. Ko, and Y. C. Chiang,

"An On-line Monitoring and Diagnostic System for Machinery of CSC No.2 Hot Strip Mill," *China Steel Technical Report*, No. 12, pp.116-126, 1998.

[2] B. Byman, T. Yarborough, R. S. V. Carolefeld, and J. V. Gorp, "Using Distributed Power Quality Monitoring for Better Electrical System Management," *IEEE Transactions on Industry Applications*, Vol. 36, NO. 5, September / October 2000.

[3] H. Wangand C. Wang, "Intelligent Agents in the Nuclear Industry," *IEEE Computer*, Vol. 30, Issue 11, pp. 28-31, November 1997.

[4] D. Lin, D. H. Zhu, F. Q. Li, and K. X. Tan, "A Distributed On-line Monitoring and Diagnosis System of Power Equipment," *Proceedings of the 6th International Conference on Properties and Applications of Dielectric Materials,* Vol. 2, 2000

[5] T. Snadholm and Q. Huai, "Nomad: Mobile Agent System for an Internal-Based Auction House," *IEEE Internet Computing*, pp.80-86, April 2000.

[6] G. Froehlich, H. J. Hoover, W. Liew, and P. G. Sorenson, "Application Framework Issues when Evolving Business Applications For Electronic Commerce," *Information Systems*, Vol. 24, No. 6, pp. 457-473, August 1999.

[7] T. Wittig, N.R. Jenningd, and E.H. Mamdani, "Archon Framework for Intelligent Co-Operation," *Intelligent Systems Eng.*, pp. 168-179, Autumn 1994.

[8] G. P. Azevedo, B. Feijó, and M. Costa, "An Agent-Based Approach to EMS in Open Environments," *Proceedings of the IEEE PowerTech, Budapest,* August 1999.

[9] G. P. Azevedo, B. Feijó, and M. Costa, "Control Centers Evolve with Agent Technology," *IEEE Computer Applications in Power*, pp. 48-53, July 2000.

[10] http://users.rcn.com/danadler/javacom/

[11] http://java.sun.som

[12] J. Hunter, *Java^{TM} Servlet Programming*, O'Reilly, ISBN 1-56592-391-X, First Edition, October 1998

[13] D. B. Langeand M. Oshima, *Programming and Deploying Java Mobil Agents with Aglets*, Addison-Wesley, ISBN 0-201-32582-9, 2nd printing, November 1998.

[14] M. Fowler and K. Scott, *UML Distilled: a brief guide to the standard object modeling language*, 2nd Edition, Addison Wesley Longman, Inc. ISBN 957-566-755-7

[15] S. A. Albir, *UML in a nutshell*, O'Reilly, ISBN 957-8247-16-8 , March 2000

[16] Z. Cui, B. Odgers, and M. Schroeder, "An In-Service Agent Monitoring and Analysis System," *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*, pp.237-244, 1999.

[17] H. Chebeane and F. Echalier, "Reactive Object Oriented Modeling of Real-time Control," *Proceedings of the 6th Internet Conference on Emerging Technologies and Factory Automation*, pp. 387-390, 1997.

[18] R. Schoop and R. Neubert, "Agent-Oriented Material Flow Control System Based on DCOM," *Proceedings of the 3rd IEEE Internal Symposium on Object-Oriented Real-Time Distributed Computing*, pp.342-345, 2000.