A Multi-Agent System for the Remote Control of Data Servers

José Henrique T. C. Sbrocco, CEATEC–Pontificia Universidade Católica de Campinas, Rod. D. Pedro I, Km 136, 13086-900 Campinas SP, Brasil; E-mail: cxhenrique@gmail.com

Ricardo Luís de Freitas, CEATEC–Pontificia Universidade Católica de Campinas, Rod. D. Pedro I, Km 136, 13086-900 Campinas SP, Brasil; E-mail: rfreitas@puc-campinas.edu.br

Gilberto S. Nakamiti, CEATEC–Pontificia Universidade Católica de Campinas, Rod. D. Pedro I, Km 136, 13086-900 Campinas SP, Brasil & Universidade Paulista, R: Dr.Bacelar 1212, São Paulo SP, Brasil; E-mail: g_nakamiti@uol.com.br

ABSTRACT

This work presents a computational tool for remotely monitoring data servers (Kipo System). To achieve this, the system contains a society of agents, where each agent performs a specific role within the process. Among the existing agents, two should be remarked: the Solver Agent, and the Supervisor Agent. The first one employs a case-based reasoning approach to interpreting and defining an action to be taken, given the detected fault. The latter one has an important role, as it constantly verifies the activation states of each agent. Through the Kipo System, a decrease in attendance times for faults on data servers could be experienced. This has corroborated our initial expectations on the adequacy of such agent's technology for this type of domain.

1. INTRODUCTION

Several software systems were developed for monitoring network behavior, as well as the operational conditions of its hardware and software components. Among the desired issues to be checked, we may include the memory available to the users, the working conditions of specific applications (such as databases), and many others. In these cases, information regarding the system's operational state, errors, and other collected data can be obtained through the analysis of log files, for example.

Many large, and even medium and small corporations have Information Technology (IT) departments, where specialized professionals are responsible for the preventive and corrective maintenance of the existing computational resources, besides other functions. They have to control the actions that identify undesired situations before they occur, avoiding unnecessary maintenance calls.

This way, the presence of identifying mechanisms, which can detect errors and undesired situations in the equipment, storage media, and data is of great importance. If the information could be retrieved remotely, without human presence, this system would be of major interest. Thus, the development of an identifying architecture which can, almost instantly, detect failures, undesired situations or potentially warning situations, is a very important issue for these corporations. As a consequence, the corporations may experience a reduction of the corrective maintenance calls, a reduction in there operational costs with personnel, and an improvement of their QoS (Quality of Service), assuring data and systems more trustable, among other gains.

This paper presents a monitoring system named Kipo[1], which uses intelligent mechanisms, designed to aid the monitoring work of geographically dispersed computers. Section 2 reviews some related works. Section 3 presents the application domain, and the next section presents the system architecture and functioning. Finally, Section 5 presents some concluding remarks.

2. APPROACHES

Since the objective was to obtain an "intelligent autonomy" regarding to the detection and sending of error information, intelligent agents characteristics were incorporated, because this technology aims at achieving some degree of "intelligence" in computer based systems [Wooldridge 95]. Solutions involving agents have been researched and applied in application areas such as air traffic control, data mining, and information retrieval, among many others [Kristensen 98].

Another research area in Artificial Intelligence (AI) is the Case-Based Reasoning (CBR), an AI paradigm mainly based on the idea that knowledge of past experiences may guide the human behavior [Kolodner 93]. This way, CBR means to use previous experiences to understand and to solve new problems.

There are some possible integrations between CBR and other approaches, such as [Marling 05, Rezende 06, Julio 06] The proposed architecture combines agents and CBR characteristics to provide intelligence with some degree of autonomy, and use of past experiences to help in the remote monitoring context.

Differently from most existing case based systems, in the proposed one, the data input will be based on parameters sent by the agents, when they detect failures. Moreover, the process of knowledge acquisition used was based on the actions that would be performed by local operators when noticing a failure.

The establishment of similarity metrics in CBR systems is one of the most important issues to achieve efficiency in such systems [Freitas 96]. Determining the similarity degree is an important component to identify the usefulness of a case. Another point to be considered is that the usefulness of a case also depends on its purposes, and which of its aspects were relevant in the past [Goel 05]. The similarity degree aims at attributing a numeric value to the similarity between two cases. All the cases stored are evaluated comparatively to the input problem (case). Usually, a similarity degree in the interval [0,1] is associated among the attributes of certain dimensions [Kolodner 93]. In the context of the architectural view presented, we used the syntactic similarity as the similarity measure. In this case, the greatest the number of coincident words between the problem description and the description of each case, the greatest the similarity degree achieved.

3. APPLICATION DOMAIN

In order to represent the use of intelligent agents and CBR in the remote corporative applications monitoring problem, it was chosen to apply them to geographically distant data servers, interconnected through a corporative network. This kind of servers usually has data servers used by several applications, and these applications need to work 24 hours a day, 7 days a week. Moreover, in the studied cases, there was only one administrator, who was responsible to guarantee the servers' operational state, as well as data integrity. Theses professionals use queries, which compute information to be compared with previously established tolerance thresholds for each monitored situation.

In the other hand, there are several professionals accessing computational resources stored in the data servers. Independently of the security policy adopted, the administrator is responsible to assure the integrity of both the data and the data structure, even if users with access rights try to modify them.

This way, if someone tries to erase a table or to modify a table's structure, this action has to be notified, so that another corrective action can be performed. Obviously, it is difficult to imagine a single person controlling all aspects of a database, without the aid of an automated system that identifies these kinds of situations, and reports them in time to avoid or minimize damage. Figure 1 depicts a general domain view of the system Kipo, exemplifying the studied situation.

Figure 1. General view of the application domain

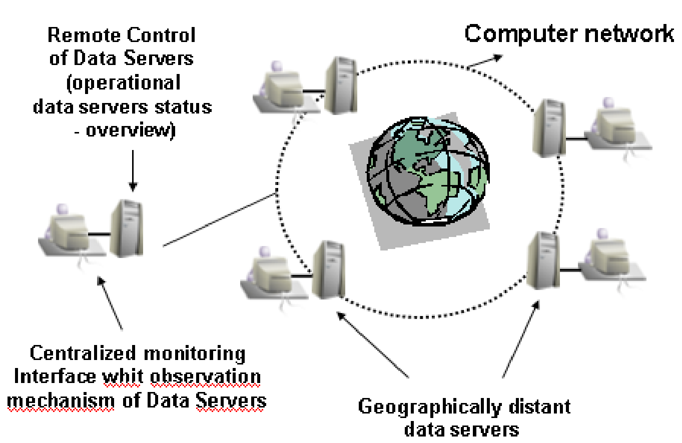
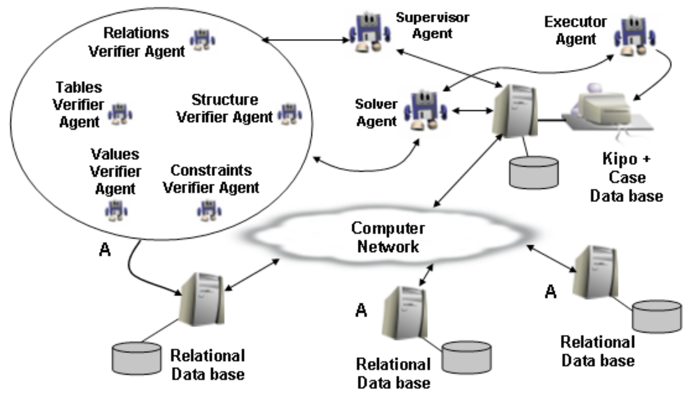


Figure 2. The main components of the Kipo architecture



Based on AI artifacts characteristics presented before, the system Kipo uses agents installed in the monitored servers, which, with a CBR agent, automatically detect and solve or inform failures and undesirable situations. An example of a failure situation could be an inactive service, the bad functioning of a specific port, the undesired change of a data base table, or even a server turned off. As examples of undesired situations, we could include a query returning an unexpected result, out of a tolerance interval, for instance. The normality standards are defined through a separated interface.

4. SYSTEM ARCHITECTURE

The motivation for the development of the system architecture included the use of agents and case-based applications for failure detection in computer networks. There are several similar works, including: Project P712 [Corley 98], SPIN [Hakima 98], PathFinder [Hart 01], SNMP [Puliafito 99], MAG [Gavalas 99], MCE [White 98], and EPSRC [Cruickshank 01], among others.

Based on these systems' behavior, the proposed model gives a new solution for the proposed domain. This way, the use of agents in the system Kipo implements alarm mechanisms, e-mail communication, and error correction, which are immediately performed after the error recognition or after a predetermined time period. These alarms can be observed through a centralized monitoring interface. This interface enables a quick and useful observation mechanism to the professional, and may be used to identify and anticipate possibly critical situations.

The Kipo architecture is composed by several intelligent agents' categories present in the data servers, each one with a specific purpose. The agents behavior and the comparing parameters they use are configured by the monitoring professional. Besides periodically perform their monitoring tasks, the agents also communicate with a supervisor agent. The detected errors are reported to a solver agent, which evaluates the situation by using CBR mechanisms. After this, it forwards the problem and its solution to an executor agent. Figure 2 depicts the main components of the proposed architecture.

The following agents' definitions detail how the system works, giving a better understanding of the architecture:

Solver Agent. This agent is responsible for the interpretation and execution of problems reported by other agents. It receives standard error situations as input, informing the sending agent, the computer identification, and the problem identified. Then, the solver agent checks for the best solution to be performed in a database, and sends a command to an executor agent. The solver agent uses CBR, checking for similarities between the reported problem and previous problems, using syntactic similarity. Figure 3 shows the agent's behavior. The arrow 1 represents the query performed by the solver agent. The arrow 2 indicates it sending an action solicitation to the executor agent, after consulting the database. The arrows with number 3 represent the possible actions to be performed by the executor agent, such as an alarm, e-mail, or error correction.

Executor Agent. This kind of agent receives information from solver agents, and they are responsible for correcting errors, sending e-mails informing about errors or updates in the database, which are related to monitoring the operational situation, among others.

Structure Verifier Agent. This agent is responsible for verifying unauthorized modifications in existing table structures. Initially, it consults the list of tables to be checked, as well as their expected structure. Then, it compares with the existing tables in the local database. Eventual discrepancies are communicated to the solver agent. Figure 4 shows this kind of agent's behavior. The correct structure of the database has to be stored and read previously, which is indicated by arrow 1. Auxiliary tables are created and used for future comparations (arrow 2). Then, periodical consults are performed in the reference database (arrow 3), and compared to the monitored tables (arrow 4). If there is any error, it is sent to the solver agent (arrow 5). The solver agent will forward the problem to the executor agent (arrow 6), which will decide if it will just report the error via e-mail, or if it will put in the monitoring interface, or if it will solve the problem (arrows 7).

Relations Verifier Agent. This agent is responsible for verifying unauthorized modifications in the predefined relations among the tables. They consult the relations of the reference database, and then, they compare them with the existing relations in the local database. Eventual differences are reported to the solver agent.

Values Verifier Agent. This kind of agent is responsible for verifying if the values get by specific queries are between predefined limits. The queries are edited using an ANSI SQL syntax interface, and unexpected results to the queries are reported to the solver agent.

Figure 3. Solver agent's behavior

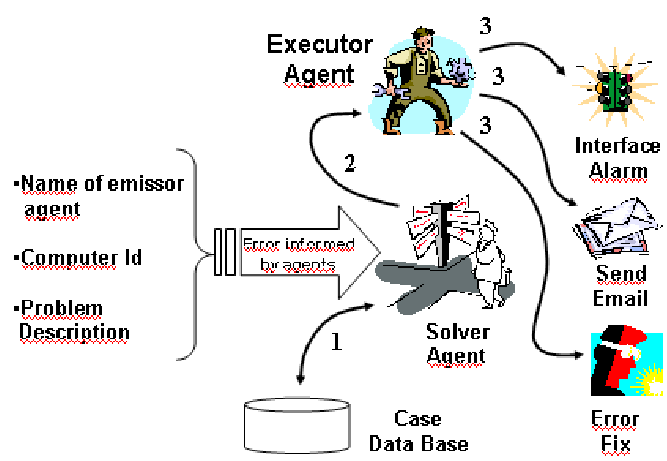
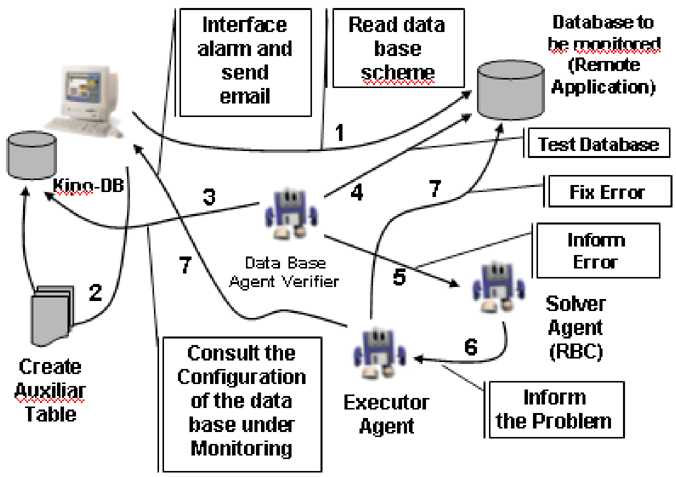


Figure 4. Structure verifier agent's standard behavior



Constraints Verifier Agent. This agent is responsible for verifying if unauthorized modifications in the predefined table constraints are performed. To do this, it consults the constraints descriptions in the reference database, and compare them with the constraints in the local database. Eventual discrepancies are reported to the solver agent.

Results Analyzer Agent. This agent compares the results got with the thresholds stored in the remote monitoring system database. Eventual discrepancies are reported to the solver agent.

Tables Verifier Agent. This kind of agent verifies if there is any table lacking in the database being monitored. To do this, it consults the list of tables in the reference database. If any table was wrongly deleted, it informs the solver agent.

Supervisor Agent. The Kipo system aims at assuring to the remote supervisor of the system a safe and trustable panorama of its software and data. Thus, safe mechanisms to verify the agents are needed. The supervisor agent supervises other agents, verifying, for example, if a server was turned off, or if it has hardware problems. The behavior of such agents is quite simple: when a remote monitoring interface is activated, its related agent is automatically activated. A signal (input parameter) identifying the sending agent is then waited. All agents, regardless of function, localization or characteristics, send periodically this type of information to the supervisor agent, assuring that they are active. A list of all agents can be consulted in the monitoring system database. Thus, if a signal is not received within a maximum amount of time, the supervisor agent updates its operational state to "without communication".

The developed architecture can be installed in any computed connected to a corporate network with databases. It uses a monitoring interface, which is responsible for the configuration of the various existing components, such as databases, information used by the agents to supervise the databases, monitoring screens, etc. An example is the interface used to check the general monitored events panorama (Figure 5), and the configuration interface, responsible for configuring the agents' behavior (Figure 6).

Figure 5a illustrates a monitored objects general state. It is based on color conventions, where the green color represents objects with normal behavior, the yellow objects means any abnormality, and red objects represents problems. Through this interface, it is possible to verify the geographical localization of the object (city, neighborhood, etc.), the description of the objects in a particular location (name, IP address, etc.), and the monitored objects list. For each object, it is possible to retrieve information about its behavior, as well as about its current operational state. The configuration interface presented in Figure 5b is used to establish each agent's behavior, regarding to their class, the periodicity of their communication with the verifier agents, their execution periodicity, and information about their function.

The agents were implemented using Java. All the agents have to consult the reference database computer (the node where the remote monitoring system is installed) to behave as established by the monitoring interface. This way, the agents need to send queries, to retrieve their syntax in the reference database, and to execute them. If necessary, they compare the query result with the values stored in the reference database. Depending on the result, they decide if they send a message to the remote monitoring system.

5. CONCLUDING REMARKS

An important contribution of this work was to explore the use of agents and CBR in the remote monitoring of computer networks domain, more specifically detecting errors and abnormal situations in distributed databases within the network. Usually, this task is performed with a human supervisor. The Kipo system auto-

Figure 5a. Remote monitoring interface

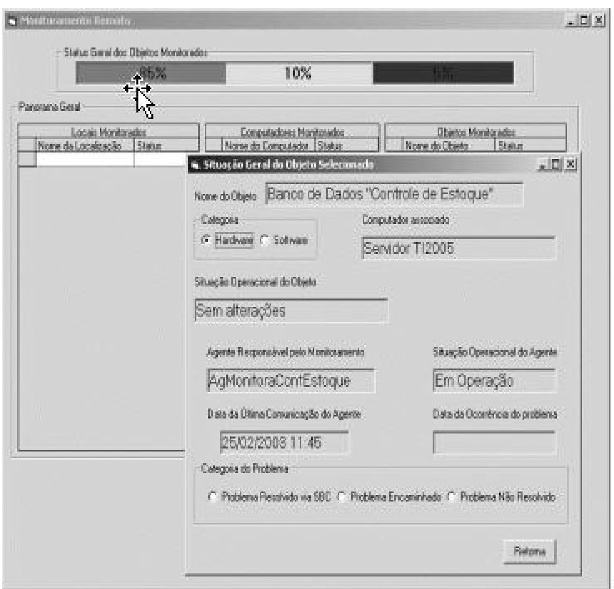
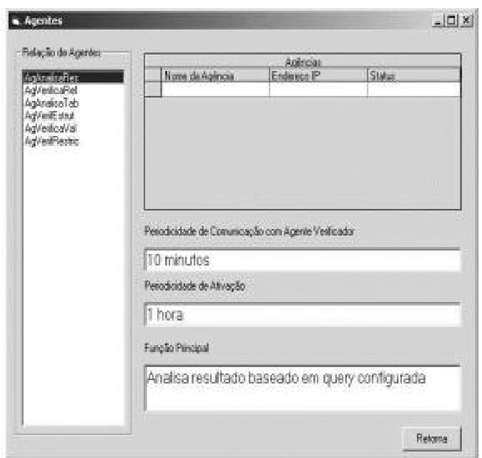


Figure 5b. Configuration interface



matically identifies and diagnoses failures, analyzes them through a case-based mechanism, signalizes the failure in an appropriate interface, automatically corrects the failure if this action is possible, and sends an e-mail describing the event occurred. Because the system tries to immediately identify failures and abnormal situations, it permits that these abnormalities to be rapidly resolved by the system agents or by another professional responsible by the monitoring system. In this context, the Kipo system is able to significantly aid to reduce the system maintenance time. Moreover, the rapid detection of failures and abnormal situations contributes to reduce the corrective system maintenance, which can be expensive to the corporations.

Currently, we are analyzing the system's first and promising results, and proposing new tests to obtain a wider set of comparative experiments. These results will be presented in a future paper.

6. REFERENCES

Corley, S., Tesselaar, M., Cooley, J., Meinköhn, J., Malabocchia, F. And Garijo, F. (1998) The Application of Intelligent and Mobile Agents to Network and Service Management. Proceedings of the Fifth International Conference on Intelligence in Services and Networks, Antwerp, Lecture Notes in Computer Science – v.1430, 127-138.

Cruickshank, D.Moreau, L.,Roure,D. (2001) Architectural design of a multi-agent system for handling metadata streams. Proceedings of the Fifth International Conference on Autonomous Agents, ACM Press, Montreal, Canada, 505-512.

Freitas, R. L., (1996) Um Sistema de Planejamento de Ações Baseado em Casos para uma célula Flexível de Manufatura, Tese de Doutorado, Escola Politécnica - USP, pp. 255, São Paulo.

Gavalas, D., Greenwood, D., Ghanbari, M., O´Mahony, M. (1999) Using Mobile Agents for Distributed Network Performance Management. Proceedings of the Third International Workshop on Intelligent Agents for Telecommunication, Stockholm, Lecture Notes in Artificial Intelligence – v.1699, 96-112.

Goel, A., Craw, S. (2005) Design, Innovation, and Case-Based Reasoning. The Knowledge Engineering Review, Cambridge University Press, UK.

Hakima, S.A., Impey, R., Liscano, R., Zeid, A. (1998) Multi-agent Testbed and an Agent Launch Tool for Diverse Seamless Personal Information Networking Applications. Proceedings of the Second International Workshop on Intelligent Agents for Telecommunication, Paris, Lecture Notes in Artificial Intelligence – v.1437, 171-189.

Hart, D., Tudoreanu, M., Kraemer, E. (2001) Mobile agents for monitoring distributed systems. Proceedings of the Fifth International Conference on Autonomous Agents, ACM Press, Montreal, Canada, 232-233.

Julio, M., Nakamiti, G., Freitas, R. (2006) A CBR Diagnostics System Applied in the Brazilian Public Health System, Proceedings of the 10th World Multi-Conference on Systemics, Cybernetics and Informatics, Orlando – FL, USA, pp. 184-189.

Kolodner, J. (1993) Case-Based Reasoning, 612 pages, Morgan Kaufman Publishers.

Kristensen, J. E. (1998) Agent Technology and Applications. Proceedings of the Fifth International Conference on Intelligence in Services and Networks, Antwerp, Lecture Notes in Computer Science – v.1430, 113-114.

Marling C., Rissland, E., AAmodt A,. (2005) Integrations with Case-Based Reasoning. The Knowledge Engineering Review, Cambridge University Press, UK.

Melchiors, C. (1999) Raciocínio Baseado em Casos Aplicado ao Gerenciamento de Falhas em Redes de Computadores, Dissertação de Mestrado, UFRGS, Porto Alegre.

Puliafito, A., Tomarchio, O. (1999) Advanced Network Management Functionalities through the Use of Mobile Software Agents. Proceedings of the Third International Workshop on Intelligent Agents for Telecommunication, Stockholm, Lecture Notes in Artificial Intelligence – v.1699, 33-45.

Rezende, M., Freitas, R., Nakamiti, G. (2006) HINT – A Hybrid Intelligent System for Telecommunication, Proceedings of the Sixth IEEE International Conference on Hybrid Intelligent Systems, Auckland, Australia.

White T., Bieszczad A., Pagurek B. (1998) Intelligent Agents for Telecommunication Applications. Proceedings of the Second International Workshop on Intelligent Agents for Telecommunication, Paris-France, Lecture Notes in Artificial Intelligence – v.1437, 130-141.

Wooldridge, M., Jennings, N. R. (1995) Intelligent Agents: Theory and Practice. The Knowledge Engineering Review, vol. 10, n° 2, 115-152.

ENDNOTE

[1] Kipo is related to a peculiar Inca communication system, which used colorful woolen cords, where knots correspond to letters and numbers. These cords were transported by messengers through the Inca Empire.