Autonomous Function in Interstage Application Server

● Hideki Nozaki

(Manuscript received November 30, 2003)

Because of the changing business environment, the ability to flexibly make configuration changes and expansions to enterprise information systems is becoming more important. Moreover, because of the large scale and complexity of modern systems, the importance of reducing the TCO (Total Cost of Ownership) has also been increasing. There is therefore a need for functions that enable simple and flexible responses to changes. One such function is autonomic computing. This is a method of ensuring simple, easy, and optimal system operation through automatic system maintenance. This paper describes an approach to autonomic computing that uses an application execution environment middleware running on Fujitsu's Interstage Application Server.

1. Introduction

To respond to developments in IT technology and changes in information system requirements, the following functions are required in the application execution environment middleware that constitutes information system platforms:

1) TCO reduction

In IDCs (Internet Data Centers) and the computer centers of large companies, multiple servers and multiple information systems are operating to support the increased load and meet other requirements. The increase in the number of servers makes it necessary to manage the work allocation of multiple servers. For example, tens of parameter settings are necessary for system construction and they must be made in each server. Also, various correction patches, modifications, and updates must be made to applications in each server. Moreover, in environments containing multiple information systems, each information system is managed independently and the overall management becomes complicated. This situation is causing the management and operation costs (TCO) in multi-system environments to increase, and a function that can reduce these costs is required.

2) Quick and easy recovery from failures

The need for information systems that operate around the clock is increasing with the advancement of services, globalization of enterprise activity, and other factors. In such systems, to avoid losing business opportunities, it is necessary to quickly detect failures and then make a quick and easy recovery. Moreover, since such systems are operating automatically, autonomous obstacle detection and recovery are required.

2. Approach of Interstage

We will now describe the present approach of Interstage.

2.1 Server consolidation and resource distribution control

One strategy for combating the increasing TCO of multiple servers is "server consolidation," which means placing all the systems under one

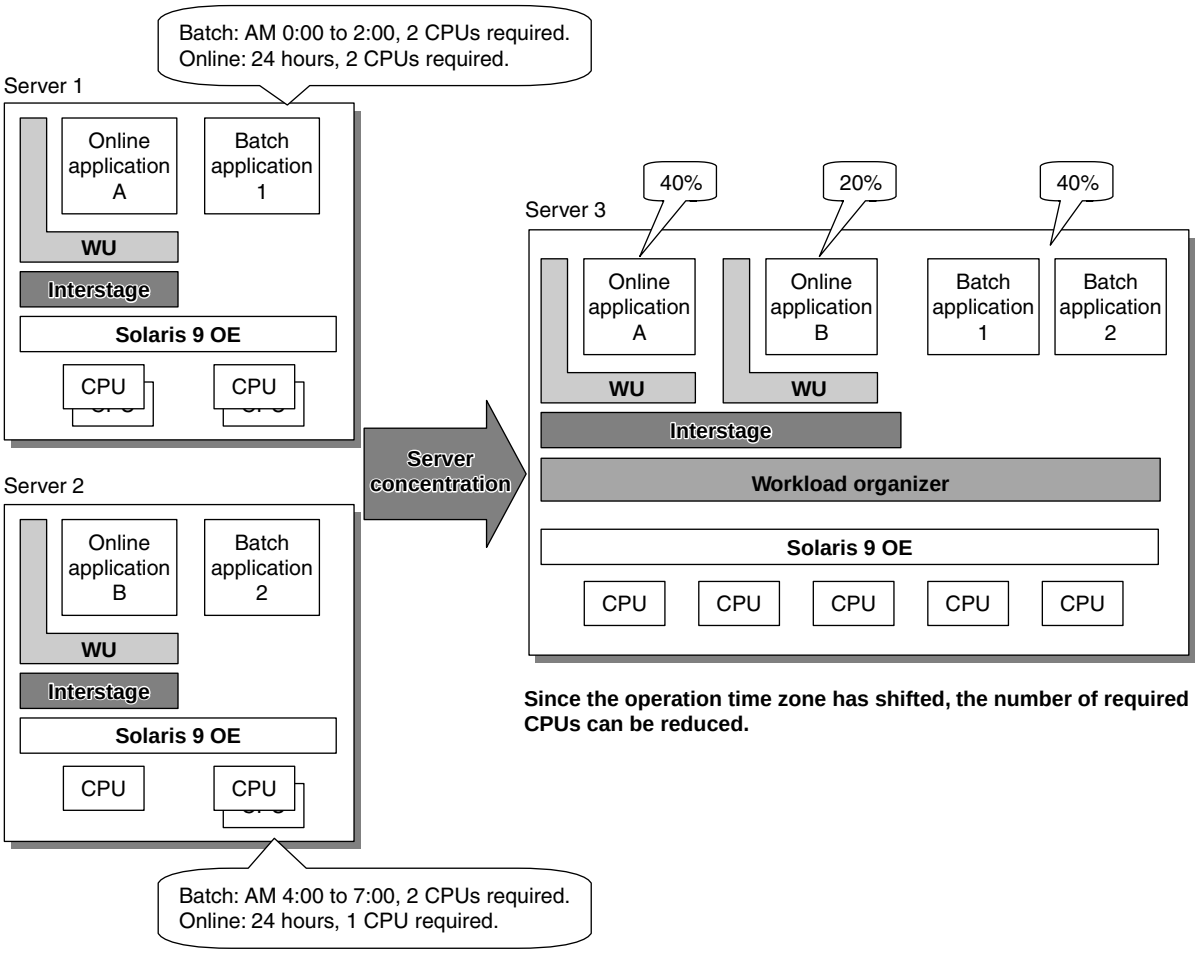

Figure 1
Server concentration and IT resource sharing.

highly efficient server. There are two effects in server consolidation.

The first is that physical costs (e.g., the cost of the space required for devices), system construction costs, and update costs can be reduced because there will only be one server. Also, OS patches only need to be applied once.

The second is that IT resource sharing can be achieved. The IT resources (CPU, network, etc.) of each information system will be prepared based on the expected peak load. If the peak loads of systems occur at different times, the total amount of IT resources being used can be reduced.

Note that server consolidation is different from conventional partitioning. Partitioning can concentrate multiple servers onto a single piece of hardware, but it is inadequate for IT resource sharing because it divides the hardware resources and operates as another system apart from the OS layer. In addition, in partitioning, the system configuration, updating, and other tasks are done separately for each system. On the other hand, since server consolidation is based on quota distribution of CPU time in process group units on the same OS, it enables flexible sharing of IT resources (**Figure 1**).

The aim of server consolidation is to provide the same service quality as when each information system has its own server. Since multiple information systems are operated on a single server, when one information system occupies an IT resource, it might not be possible to maintain a sufficient service level. In Interstage, this problem is solved by the workload management

function (Workload Organizer) that operates on Solaris 9 OE and providing a server consolidation solution for information systems. The workload management function makes a group of processes and controls the allocation of CPU time between groups. In the workload management function, this group is called a resource module. Interstage manages application processes in work units (WUs). In order to start and stop applications in WUs, applications are managed and operated in WUs. Then, the system administrator can allocate CPU time from the WU point of view by associating WUs with modules.

In this way, multiple information systems can be operated without individual systems monopolizing CPU time. For example, if WU1 and WU2 are allocated, respectively, 60% and 40% of the total CPU time, even if their loads increase, each WU still has sufficient CPU resources. Also, the CPU resources can be divided between applications running on Interstage and applications running on other servers. For example, it is possible to divide CPU resources between an online application running on Interstage and a batch application.

Moreover, by using the workload management function, the total amount of required IT resources can be reduced by changing the distribution of allocated CPU time, for example, when information systems have different peak-load times. The workload management function makes it easier to change the CPU time allocation, because it allows it to be done without system termination.

2.2 Autonomic trouble detection, restoration, and prevention

In an information system for 24/7 operation, application stops due to abnormalities can result in lost business. Therefore, it is important to quickly detect abnormalities and restore the system. Interstage provides an autonomic detection and restoration function for application abnormalities.

1) Recovery from abnormally terminated applications

Interstage manages and monitors application processes in WUs. The WU monitoring function automatically detects abnormal terminations and then restarts the affected application. This function minimizes the influence of abnormal terminations on information systems.

2) Automatic restoration of hung applications

The WU monitoring function handles hung applications in the same way it handles abnormally terminated ones. When an application hangs, it is forcibly terminated and then automatically restarted. This function minimizes the influence of hung applications on information systems.

Figure 2 shows recovery from abnormally terminated and hung applications.

3) Sign monitoring

To prevent trouble in an information system, the behavior of applications and the system is monitored and a function for detecting and reporting the signs of potential trouble is provided.

There are two functions in sign monitoring. The first monitors the memory of JavaVMs. When Java is used for Servlet/EJB application development and the amount of transactions is increased, some transactions might not be executed because of a memory shortage and performance may be slowed down due to frequent garbage collections (GCs). Interstage therefore provides a function for preventing response deterioration due to GCs by managing the memory of JavaVMs. Interstage outputs reports of alarm messages, memory shortages, GCs, and other events occurring in the JavaVMs that can cause slowdowns. This enables system administrators to increase the number of JavaVMs being used and operate information systems appropriately.

The second function monitors the number of waiting requests. The number of requests from clients that are waiting to be processed by a server application is monitored, and if the number of requests exceeds a fixed number, subsequent

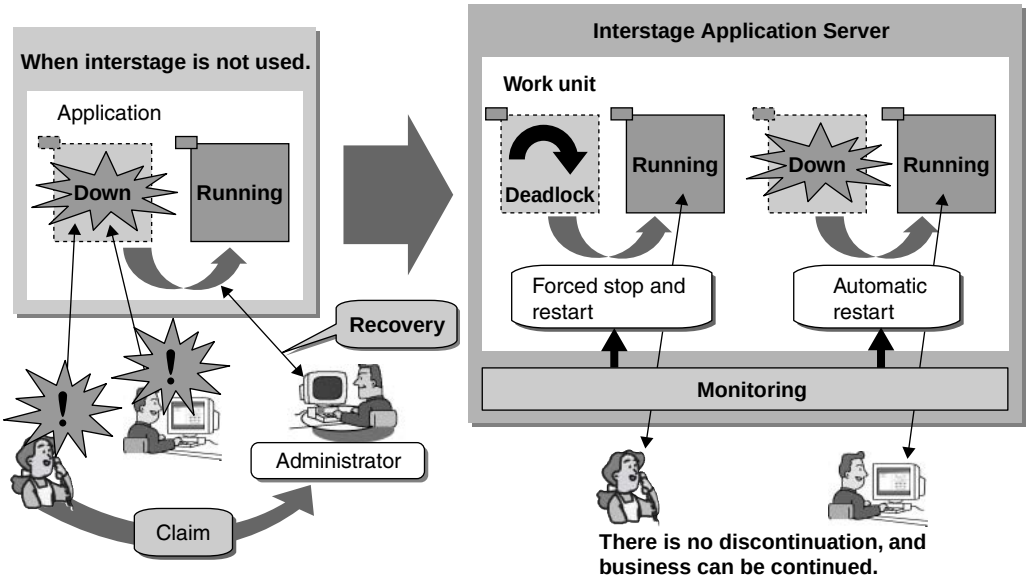

Figure 2
Minimizing influence by work unit automatic restart.

requests will be refused. Also, a function for preventing the generation of more than a fixed number of waiting requests is provided. Therefore, the response to clients is guaranteed and server IT resources can be saved. Moreover, abnormal symptoms are reported to the system administrator. These functions make it possible to tune a system according to the current conditions.

4) Automatic expansion of processes and thread execution

In online transaction processing (OLTP) information systems, the processing performance is improved by increasing the concurrency of processes and threads. However, starting processes and threads consumes IT resources such as memory. In Interstage, thread concurrency is increased automatically according to the amount of client requests, and when a temporary peak in the load ends, the function stops additional threads. These functions are carried out automatically, so the appropriate concurrency for the load is achieved without wasting IT resources.

A process concurrency change function that cooperates with the workload management function is also provided. The workload management function adjusts the process concurrency according to the processing status and the amount of CPU time being used. If a request is waiting to be processed and there is free CPU time, the concurrency is insufficient. In this case, the function increases the concurrency and the CPU processes more requests. These control functions are achieved through a previously set cooperation between Interstage and the workload management function.

3. Conclusion

Interstage aims at achieving autonomous computing. Currently, Interstage is an application server that provides autonomous obstacle detection/restoration functions to operate complicated information systems and reduce the TCO. An information system can be constructed without developing special operation management functions by using the autonomy functions provided by Interstage.

For the next or a later version, we plan to add extra functions, for example, a non-server-consolidation method of operation for multi-server environments. When an increase in load cannot be estimated, a system can be designed to accom-

modate an additional server when it is required. Furthermore, in terms of technology and cost, it is easier to construct multiple servers than before due to the popularization of Blade servers. Because of these considerations, we can expect an increase in the number of load-distributed, multi-server systems and we believe the need for simple, multi-server operation will increase.

Moreover, in the near future, information systems with server groups on different types of platforms (OSs) are expected to increase because of progress in grid technology. Therefore, functions for handling such systems will be required.

As a result, the following enhancements will be made to the autonomy functions of Interstage to provide a superior autonomic computing environment.

1) Provisioning function

To ensure scalability and effective use of IT resources, multiple servers will be virtually pooled using a provisioning function. This function will lend a server from the pool to an information system that requests it according to the load change and service level. The workload management function can be used in single-server and multi-server environments. Interstage will provide a function to obtain additional servers when they are needed to maintain the service level of each WU and then return them to the pool when they are no longer required.

For example, assume the case of an environment containing four pooled servers. The server load is high during the day, so the system is configured with three Web servers and one application server. However, at night, the number of business transactions on the application server increases, so the system is then configured with one Web server and three application servers. In this way, the provisioning function facilitates easy, dynamic configuration changes to suit the load level.

2) Operability improvement in multiple servers: Single System View

Currently, the main advantages of server consolidation, which are simplicity of system construction and modification, will not be achieved in multi-server systems. Therefore, the same operability as that of a single server must be provided. We are currently developing an Interstage function for building, modifying, and operating WUs running on multiple servers. This function will enable system administrators to operate WUs without considering the number of servers.

3) Enhancement of the trouble prevention function

The trouble prevention function based on sign monitoring will be enhanced. Currently, this function only looks for signs and reports them to the system administrator. However, in the next or a later version, when a sign is detected, the load will be distributed to alternative JavaVMs and a function to automatically prevent the generation of abnormal transactions due to, for example, an insufficient JavaVM heap area, will be provided.

Hideki Nozaki received the B.S. degree in Electrical Engineering from Waseda University, Tokyo, Japan in 1988. He joined Fujitsu Ltd., Kawasaki, Japan in 1988, where he has been engaged in development of middleware software.