G00002

DESCRIPTION

MOST ELIGIBLE SERVER IN A COMMON WORK QUEUE ENVIRONMENT

BACKGROUND OF THE INVENTION

1. TECHNICAL FIELD

The present invention relates generally to parallel processing environments, and more specifically to parallel processing systems employing a shared queue.

2. BACKGROUND ART

It is commonplace in contemporary data processing environments to provide a plurality of systems to handle the processing needs of one or more clients. For example, two or more systems, such as transaction processing systems, may be interfaced to one or more clients via a communications network. In this environment, when a client has a task to be performed by one of the systems, that client sends an input message to the desired system to request processing by an application running in that system. The subject system queues the message and provides the message to the application for processing. When processing is complete, the application places an outgoing message in the queue for transmission over the network to the client.

To take advantage of the multi-processing aspect of this environment, the system originally tasked by the client, system A, may extract the input message from its queue and forward the input message to a second system, system B, for processing. When processing is completed by system B, the response (outgoing message) is forwarded to system A and placed on system A's queue for transmission to the client. Thus, in this manner, multiple systems can be utilized to handle processing requests from numerous clients.

There are, however, a few disadvantages with this arrangement. For example, if system A fails, none of the work on the queue of system A can be accessed. Therefore, the client is forced to wait until system A is brought back online to have its transaction processed.

In order to address these disadvantages, a shared, or common, queue may be provided to store incoming messages for processing by any of a plurality of data processing systems. A common queue server receives and queues the messages onto the shared queue so that they can be retrieved by a system having available capacity to process the messages. In operation, a system having available capacity retrieves a queued message, performs the necessary processing, and places an appropriate response message back on the shared queue. Thus, the shared queue stores messages sent in either direction between clients requesting processing and the data processing systems that perform the processing.

Because the messages are enqueued onto the shared queue, the messages can be processed by an application running in any of a plurality of systems having access to the shared queue. Thus, automatic workload management among the plurality of systems is provided. Also, because any of the systems connected to the shared queue can process messages, an advantage of processing redundancy is provided. If a particular application that is processing a message fails, another application can retrieve that message from the shared queue and perform the processing without the client having to wait for the original application to be brought back on-line. This provides processing redundancy to clients of the data processing environment.

When utilizing a common work queue, all clients that submit requests onto the common queue have an expectation that their work request will be completed within a reasonable amount of time. In an environment where not all servers are eligible to

process all requests, it is possible at any given time that there are no servers eligible to process some or all of the work requests on the queue. When a work request is synchronous, it is important that the client that submitted the request be notified that there is no server eligible to process its synchronous request.

More specifically, in a shared work queue environment, there are one or more clients that submit requests and one or more servers that process those requests. Of those servers that process requests, not all servers may have the ability to process each request on the common work queue. This may be the result of (a) a server has been configured to only process certain types of requests or (b) a server does not have access to the resources required to process certain types of requests. As an example for case (a), some servers may be configured to only process requests from certain clients, while other servers may be configured to process all requests. As an example for case (b), some requests may require access to certain storage pools that not all servers have connectivity to. Only servers that have connectivity to the required storage pool can process those requests.

In CA 2 282 931 A (MICROLOG CORP), „A CONTACT CENTER SYSTEM CAPABLE OF HANDLING MULTIPLE MEDIA TYPES OF CONTACTS AND METHOD FOR USING THE SAME", a contact center system and method employing a plurality of agent workstations is introduced, which includes a queuing component, capable of receiving contacts of different media-types, and which maintains the different media type contacts in a common queue. The system further comprises a routing component which routes the queued contacts to the agents based on criteria of the contacts, criteria of the agents, or both.

In WO 94 20904 A (ROLM CO), „QUEUE MANAGING SYSTEM AND METHOD", a queue management system for servicing of a number of clients representing different client types is introduced.

A queue manager places the clients into a processing queue matching the client type. In the environment of a telephone system the clients are messages and the client types are codings in the messages for various destinations.

In EP 1 126 371 A (SONY CORP ; INFORMATION BROADCASTING LAB I(JP)), „Service providing apparatus transmitting apparatus, receiving apparatus, and receiving method", a client-server system is introduced broadcasting a directory of services to clients. A client searches its received directory of services to select a server providing a service.

In US 5 887 168 A (PATTERSON BETTY J ET AL), „COMPUTER PROGRAM PRODUCT FOR A SHARED QUEUE STRUCTURE FOR DATA INTEGRITY", a shared queue is provided to allow any of a plurality of systems to process messages received by the clients. Any of the plurality of systems having available processing capacity can retrieve the message from the shared queue and process the message. A response to the message, where appropriate, is enqueued onto the shared queue for delivery back to the client. A unique list structure is provided to implement the queue. The list structure is comprised of a plurality of sublists, or queue types.

SUMMARY OF THE INVENTION

An object of this invention is to improve data processing systems as the one of US 5 887 168 A, that use a common queue to process work requests by servers adapted to process some or all types of requests, by maintaining the status of eligibility for each of the different types, that is, whether at least one of the servers is eligible for processing of said each of the different types.

Another object of the invention is to notify a client, who has submitted a synchronous request, whether there is a server eligible to process that request allowing this client to

remove its request from the shared queue in case there is no server eligible to process its request.

A further object of the invention is to prevent a synchronous request from being submitted for which there is no server eligible to process.

These and other objectives are attained with a system ~~and~~ , **a** method **and a program** for managing servers **provided by the invention according to claims 1, 9 and 13.** The system comprises a plurality of servers for processing messages, at least one client for transmitting messages to and receiving messages from the servers, and processor means for maintaining a list indicating whether at least one of the servers is eligible to process messages. Preferably, the servers are adapted to process messages of different types; and the processor means includes means to indicate on said list whether, for each of the different types of messages, at least one of the servers is eligible to process each of the different types of messages.

Also, in the preferred embodiment, each of the clients includes means to read the list, before the client transmits a message of a given type to one of the servers, to determine whether at least one of the servers is eligible to process messages of the given type. In addition, the processor means may notify the clients when changes are made to the list, and each of the clients may include means for removing messages from the common queue in case there is no longer any server eligible to process the messages. A queue server provides the necessary interface for the common queue. Any suitable queue server may be used, however, preferably the common queue server is the Coupling Facility available in the zSeries processing environment available from International Business Machines, Corp.

6.06.2004

29 AUG 2006

EP0302674

Further benefits and advantages of the invention will become apparent from a consideration of the following detailed description, given with reference to the accompanying drawings which specify and show preferred embodiments of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram showing a shared queue in a client/server environment.

AMENDED SHEET

Figure 2 shows a shared queue processing environment in which a most eligible server status record is used.

Figure 3 illustrates a procedure for updating a most eligible server status record.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention generally relates to systems and methods that may allow any of a plurality of processing systems to process messages for one or more clients. In the preferred embodiment, a structured external storage device, such as a shared queue, is provided for queuing client messages for the plurality of systems. When incoming messages are received from the clients, they are placed on the queue. When one of the plurality of systems has available processing capacity, it retrieves a message, processes the message and places a response on the queue.

Figure 1 is a block diagram illustrating the shared queue in a client/server environment 10. The client/server environment includes one or more clients 12 interfaced to a plurality of processing systems 14 via one or more networks 16. When a client 12 has a transaction to be processed, the client enqueues the message onto shared queue 20. As additional messages are received from clients, they too are enqueued onto the shared queue. Each message remains on shared queue 20 until it is retrieved by one of the systems 14 for processing.

When a system 14 determines that it has the capacity to process another transaction, that system 14 dequeues a message from shared queue 20. That system 14 then processes the message and places on shared queue 20 the appropriate response to the client that generated the incoming message. A common queue server 22 provides the necessary interface between shared queue 20 and systems 14. When an input message is received by the common queue server for enqueueing onto shared queue 20, the queue server 22 buffers the message in one or

more buffers and then transfers this data to the shared queue. Any suitable common queue and common queue server may be used in the practice of this invention. Preferably, though, the common queue server is the Coupling Facility available in the zSeries processing environment available from International

Business Machines, Corp. This Coupling Facility provides various features used in the preferred implementation of the invention.

As mentioned above, one difficulty that can occur when a common work queue is used is that it is possible that at any given time, there are no servers eligible to process some or all of the work requests on the queue. To address this need, this invention utilizes a feature referred to as the "Most Eligible Server." The most eligible server is that server that has the greatest ability to process requests. By knowing the status of the most eligible server, clients may submit requests to the common work queue based on whether or not there is a server that is eligible to process that request.

In order to prevent a synchronous request from being submitted for which there is no server eligible to process, a "Most Eligible Server Status" (MESS) record is maintained. The MESS record indicates whether or not there is a server eligible to process requests, and if so, which types of requests are eligible to be processed. The number of servers eligible does not need to, but can, be maintained. The MESS record indicates that there is at least one server eligible to process requests, so that requests do not remain on the queue for an indefinite amount of time. Before submitting a request, a client quickly reads the MESS record to determine if there is a server eligible to process its request. If so, the client then places its request onto the queue.

Preferably, if there is a change in the MESS record, then clients are notified of the change. If the MESS is lower than its previous status - that is, no server is eligible or certain types of requests can no longer be processed - then the client has the option of removing its requests from the queue for which there is no longer a server eligible to process.

Figure 2 illustrates an environment in which a most eligible server status record may be used. As an example, two servers 32, 34 are processing requests from a common work queue 36. Both Servers are connected to a Disk Pool 40, but only one of the servers can be connected to the Tape Pool 42. Server 34 is the only server that can process requests that require a Tape Resource. As long as Server 34 is available, the MESS record indicates that there is a server available to process requests requiring tape. When Server 34 is not available, then the MESS record 44 is updated to indicate that there are no servers available to process tape requests, and Clients 46 and 50 can fail existing outstanding tape requests and submit new requests accordingly.

Figure 3 shows a procedure for updating a most eligible server status record. As represented at steps 60 and 62, when a server connects to the common work queue, it determines if its eligibility to process requests is greater than is currently reflected in the MESS record. If so, then at step 64 the MESS record is updated; and if not, then, as represented by step 66, no change is made in the MESS record. The record may also be updated when there is a change in eligibility of the connected servers. A record of the eligibility status of each server may be maintained in order to make this determination. When a server disconnects from the common work queue, then the MESS record may be updated if the status of the MESS has changed.

The processing needed to maintain and update the MESS record may be done by any suitable processor means. For instance, the queue server 22 may be used to maintain and update this list, one or more of the processing systems 14 may be used for this purpose, or a separate device may be provided to perform these functions. Also, depending on the specific environment in which the present invention is employed, this processor means may include a single processor or plural processors. For instance, depending on the specific system in which the invention is used, the MESS record may be maintained and updated by a standard personal computer having a single processing unit, or by any other suitable type of computer, including, for instance, computers having plural or multiple processor units. Further, it may be noted, the needed processing may be done principally by software, if desired, principally by hardware, or by a combination of software and hardware.

Claims:

1. A server management system comprising:

at least one client for transmitting messages to and receiving response messages from a shared queue;

a plurality of servers for processing client messages from the shared queue and transmitting response messages to the shared queue, the servers being adapted to process messages of different types;

the system being characterized by:

a list indicating for each of the different types of messages, whether at least one of the servers is eligible to process such different type of message at any time.

2. A server management system according to Claim 1, additionally comprising a common queue server wherein the shared queue is maintained.

3. A server management system according to Claim 2, wherein the list is maintained at the common queue server.

4. A server management system according to Claim 1, wherein the list is maintained at one or more of the servers.

5. A server management system according to Claim 1, wherein the list is maintained at a separate device.

6. A server management system according to Claim 1, wherein each of the clients includes means to read said list, before the client transmits a message of a given type to the shared queue, to determine whether at least one of the servers is eligible to process messages of the give type.

7. A server management system according to Claim 1, wherein the processor means notifies the clients when changes are made to said list.

8. A server management system according to Claim 1, wherein each of the clients includes means for removing messages from the shared queue in case there is no longer a server eligible to process messages of a given type.

9. A method of managing servers consisting the steps of:

using at least one client to transmit messages to and to receive response messages from a shared queue;

providing a plurality of servers for processing client messages from the shared queue and transmitting response messages to the shared queue, the servers being adapted to process messages of different types;

providing a list indicating for each of the different types of messages, whether at least one of the servers is eligible to process such type of message at any time.

10. A method according to Claim 9, further comprising the step of, each of the clients reading said list, before the client transmits a message of a given type to the shared queue, to determine whether at least one of the servers is eligible to process messages of the given type.

11. A method according to Claim 9, further comprising the step of using the processor to notify the clients when changes are made to said list.

12. A method according to Claim 9, further comprising the step of, each of the clients removing messages from the shared queue in case there is no longer a server eligible to process messages of a given type.

13. A program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform method steps for managing a plurality of servers according to the preceding claims.