Computation & Communication in R*: A Distributed Database Manager

by

Bruce G. Lindsay, Laura M. Haas, C. Mohan, Paul F. Wilms, & Robert A. Yost

IBM San Jose Research Laboratory
5600 Cottle Road
San Jose, CA 95193

EXTENDED ABSTRACT

R* is an experimental prototype distributed database management system. The computation needed to perform a sequence of multi-site user transactions in R* is structured as a *tree* of processes communicating over virtual circuit communication links. Distributed computation can be supported by providing a server process per site which performs requests on behalf of remote users. Alternatively, a new process could be created to service each incoming request. Instead of using a shared server process or using the process per request approach, R* creates a process associated with the computation of the user on the first request to the remote site. This process is incorporated into the tree of processes serving a single user and is retained for the duration of the user computation. This approach allows R* to factor some of the request execution overhead into the process creation phase, and simplifies the retention of user and transaction context at the multiple sites of the distributed computation.

R* uses virtual circuit communication links to connect the tree of processes in an R* computation. Virtual circuits provide message ordering, flow control, and error detection and reporting. Especially important in the distributed transaction processing environment is the ability of the virtual circuit facility to detect and report any process, processor, or communication failures to the end points of the virtual circuit. Error detection and reporting by the virtual circuit facility is used to manage the tree of processes comprising a user computation and to handle correctly the resolution of distributed transactions in the presence of various kinds of failures.

R* uses the communication facility in a variety of ways. Many functions use a syn-

chronous, remote procedure call protocol to perform work at remote sites. Site authentication, user identification, data definition, and database catalog management are all implemented using this remote procedure activation protocol. Query planning, on the other hand, distributes query execution plans in parallel to the sites involved. Parallel plan distribution allows server sites to overlap the computation needed to validate and store query execution plans. The query execution plans often involve passing data streams from site, to site with each site transforming the data stream in some way. The execution of data access requests exploits virtual circuit flow controls to allow overlapped execution at data producer and data consumer sites.

Finally, distributed transaction management in R* uses the virtual circuits connecting the process tree to exchange the messages of the two-phase commit protocol. However, if a failure occurs during the commit protocol, the virtual circuits and process of the original computation may be lost. When failures interrupt the distributed commit protocol, R* reverts to a datagram-oriented protocol to transfer the messages needed to resolve the outstanding transaction. R* also uses datagrams to communicate the information needed to detect multi-site deadlocks.

The R* approach to distributed computation may be contrasted with datagram-based and server-oriented distributed systems. The retention of remote processes, and the virtual circuits connecting them, for the duration of the user computation improves execution performance whenever repeated accesses are made to a remote site. The retention of remote processes is also helpful for maintaining user and transaction context between requests to the remote site. The use of virtual circuits allows several concerns, such as message ordering and flow control, to be relegated to the network and virtual circuit implementation. The ability of the virtual circuit implementation to report failures is fundamental to the management of the R* distributed computation.

Currently, R* is running on multiple processors and is able to perform any SQL statement on local or remote data. This includes not only data definition and catalog manipulation statements, but also n-way joins, subqueries, and data update statements. Besides the SQL language constructs, the transaction management and distributed deadlock detection protocols are implemented and running. The tree structure of the R* computation and the use of virtual circuits have proved to be quite well adapted to the problems of implementing and controlling the complex, distributed computations needed to support the execution of a distributed database management system.