# 6 Early Bottlenecks

Prior to the advent of distributed computing, just about all mission-critical application programs revolved around IBM systems. Eventually, the need for application resources outstripped even the largest IBM machine, leading to the concept of the TPM.

One of the first transaction processing systems was IBM's CICS, which continues to be a leading transaction processing product today. CICS software allows system resources such as memory and CPU cycles to be allocated to the appropriate programs waiting for terminal input or disk I/O rather than some external event. TPMs can not only extend the processing capability, they can bracket the updating of resources like files and databases within the concept of a transaction. In other words, a program can control when a transaction is initiated and terminated and when and which resources are updated or rolled back. Similar technology is incorporated in IBM's IMS and Tandem's Pathway and TMF products.

For more than two decades, scalable mission-critical applications emanated from IBM and Tandem mainframes, with accessibility provided by non-intelligent screen devices; this was the "dumb terminal." Application complexity, high cost of deployment or specialized programming skills were not issues—this was state-of-the-art computing and there were no viable alternatives.

The advent of personal computers and distributed computing over the past 15 years has written a separate history of alternative programming environments, which application developers prefer for their low cost, speed of development and higher quality user experience. Before you write off the mainframe TPM as a Jurassic Park exhibit, however, understand that, in terms of performance, the TPM is unequaled with:

- More highly scalable systems
- Increased data integrity
- Better performance tuning and load balancing
- Superior resource management

- More efficient resource utilization.

If, in fact, the application server is on its way to becoming the modern equivalent of a TPM, with the advantage of host independence, you can look forward to an exciting set of innovations to enterprise computing.

Application servers offer server-side support for developing and deploying business logic -- business logic that may be located on the server or, more often, partitioned across client and server. This is nothing new: Enterprises rely daily on server-side business processing, ranging from mainframe transaction systems to client/server DBMS stored procedures. Running business processes on the server provides the following:

- *Re-use.* A variety of client applications (HTML-only, Java applets, COM+ components, etc.) can share the same business logic.
- *Intellectual property protection.* Sensitive business logic often includes or manages trade secrets that could potentially be reverse engineered.
- *Security of business logic.* By leaving the logic on the server, user access can be controlled dynamically, revoked at any time.
- *Security of network communications.* Application servers allow use of internet-standard secure protocols like SSL or HTTPS in place of less secure proprietary DBMS protocols.
- *Manageability.* Server-side applications are easier to monitor, control, and update.
- *Performance.* Database intensive business logic will often perform much better when located near the database, saving network traffic and access latency.
- *Download time.* I*Net (intranet + extranet + internet) clients most often require access to many different business processes that could require substantial network bandwidth and client memory to download all logic to the client.
- *Compute load.* Running compute-intensive applications on servers saves client cycles.

With the explosive growth of the I*Net, there is unfulfilled demand for application server technology to accompany the now pervasive web infrastructure -- ***application server technology that goes well beyond CGI and its successors.***