There is still much about the Darkstar architecture that we have not tested or that we don't fully understand. Although we have produced a system that allows multiple machines to run a game or virtual world utilizing multiple threads in a way that is (mostly) transparent to the server programmer, we have not yet tested the ability of the architecture to add other services beyond the core. Given the transactional nature of Darkstar tasks, this may turn out to be more complex than we first imagined, and our hope is that the additional services will not need to be participants in the core service transactions. We have also just begun to experiment with various ways of gathering information about the load on the system and balancing that load. Fortunately, since the mechanisms that do this balancing are completely hidden from the programmers using the system, we can pull out old approaches and introduce new ones without affecting those using Darkstar.

As an architecture, Darkstar presents a number of novel approaches that make it interesting. It is one of the few attempts to build a game or virtual world infrastructure with the same reliability and dependability properties as enterprise software while also meeting the latency, communication, and scaling requirements of the game industry. By trying to gain efficiency by using more machines and more threads, we hope to offset the increases in latency we introduce by the use of a persistent storage mechanism. Finally, the very different world of games and virtual environments, in which the clients are thick and the servers are thin, presents a contrast to the usual environment in which highly concurrent, distributed systems are generally built. It is too early to tell whether the architecture is going to be successful, but we believe that it is already interesting.

Download at WoweBook.Com