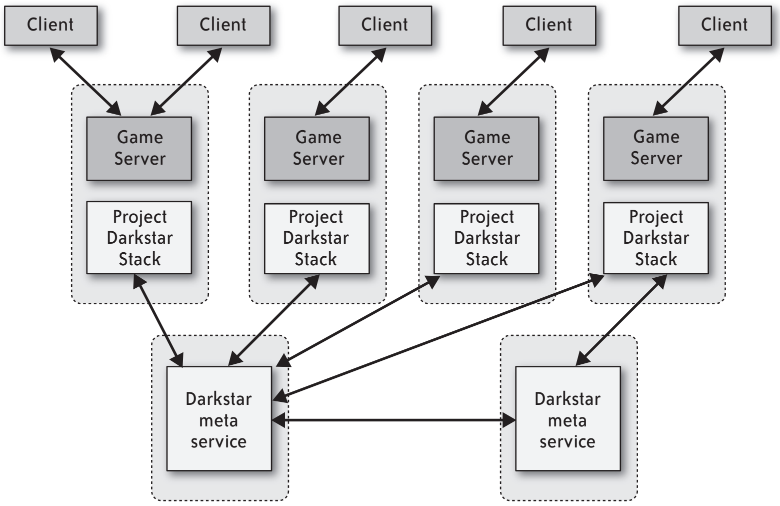

FIGURE 3-1. Project Darkstar high-level architecture

Unlike most replication schemes, the different copies of the game logic are not meant to process the same events. Instead, each copy can independently interact with the clients. Replication in this design is used primarily to allow scale rather than to ensure fault tolerance (although, as we will see later, fault tolerance is also achieved). Further, the game logic itself does not know or need to know that there are other copies of the server operating on other machines. The code written by the game programmer runs as if it were on a single machine, with coordination of the different copies done by the Project Darkstar infrastructure. Indeed, it is possible to run a Darkstar-based game on a single server if that is all the capacity the game needs.

Clients connect to the game logic using communication mechanisms that are part of the infrastructure. These mechanisms allow either direct client-to-server communication or a form of publish-subscribe channel, where any message sent on a channel is delivered to all of those subscribed to the channel.

The Darkstar stacks are coordinated by a set of meta-services—network-accessible services that are hidden from the game or virtual world programmer. These meta-services allow the various copies of the stack to coordinate the overall operation of the game. These meta-services will, for example, make sure that all of the separate copies continue to run and initiate failure recovery if some copy fails; keep track of the load on the copies and redistribute that load when needed; or allow new servers to be added at any time to increase the capacity of the whole. Since these services are completely hidden from the users of Project Darkstar, they can be changed or removed, or new ones can be added at any time without changing the code of the game or virtual world.