Abstract

Essentially all distributed systems, applications, and services at some level boil down to the problem of managing distributed shared state. Unfortunately, while the problem of managing distributed shared state is shared by many applications, there is no common means of managing the data—every application devises its own solution. We have developed Khazana, a distributed service exporting the abstraction of a distributed persistent globally shared store that applications can use to store their shared state. Khazana is responsible for performing many of the common operations needed by distributed applications, including replication, consistency management, fault recovery, access control, and location management. Using Khazana as a form of middleware, distributed applications can be quickly developed from corresponding uniprocessor applications through the insertion of Khazana data access and synchronization operations.

1 Introduction

Essentially all distributed systems applications at some level boil down to the problem of managing distributed shared state. Consider the following application areas:

- Distributed file systems (AFS, NFS, NTFS, WebNFS, CIFS, ...)
- Clustered file systems (DEC, Microsoft, ...)
- Distributed directory services (Novell’s NDS, Microsoft’s Active Directory, ...)
- Distributed databases (Oracle, SQL Server, ...)
- Distributed object systems (DCOM, CORBA, ...)
- Collaborative groupware (Lotus Notes, Microsoft Exchange, ...)

All of these services, and many more, perform essentially the same function, albeit in very different settings. That function is managing distributed shared state and providing a convenient way for users and other applications to access, update, and delete the information being managed. Unfortunately, while the problem of managing distributed shared state is shared by all of the above applications, there is no common infrastructure for managing shared data, so every system implements its own solution. The thesis of this paper is that it should be possible for distributed clients and servers to share state without each instance of sharing requiring specially written code. Just as TCP/IP hides many complex issues from programmers (e.g., handling link failures, routing, and congestion), there should be support for distributed state sharing that lets most applications remain oblivious to the many problems associated with managing shared state (e.g., heterogeneity, security, high availability, caching strategies, and coherence management). We are developing Khazana, a distributed service to provide this support.

Figure 1 illustrates a typical Khazana-based distributed system consisting of five nodes. Applications such as those described above can use Khazana to store shared data that can be accessed from any node connected to Khazana. The Khazana design assumes that some or all of the nodes may be connected via slow or intermittent WAN links. Khazana will use local storage, both volatile (RAM) and persistent (disk), on its constituent nodes to store data. In the example illustrated in Figure 1, the square represents a single piece