# Chapter 4.4
# Large-Scale ASP Replication of Database-Driven Portals

**Christopher B. Mayer**
*Air Force Institute of Technology, USA*

**K. Selçuk Candan**
*Arizona State University, USA*

## INTRODUCTION

Web portal applications that dynamically generate results in response to user requests are more popular than ever. Such portal applications usually consist of a business logic component and a very large database, or databases that hold the portal's content. Despite efforts to speed up response generation, ever-rising user demand means that replication of a portal's logic *and* database will be needed at some point as other methods to keep up with demand (faster databases and content caching for example) have limits.

This article explains issues surrounding the replication of a single full-scale database-driven portal application. In addition to the issues of replicating a single, unsophisticated application, it also anticipates a future in which portal applications offer multiple levels of service to users, and large application service providers (ASPs) host and replicate many portal applications on networks of servers. An ASP must replicate complex portal applications in order to satisfy user demand while minimizing operating costs. ASPs will need mature tools for making replication decisions and deploying and otherwise managing their replication system. To that end, this article highlights two prototype software packages, ACDN and DATE/DASIM, that address some aspects of replica management by ASPs. Using the two prototypes as a starting point and recalling single portal replication issues, a set of features for a mature ASP replication management systems are proposed.

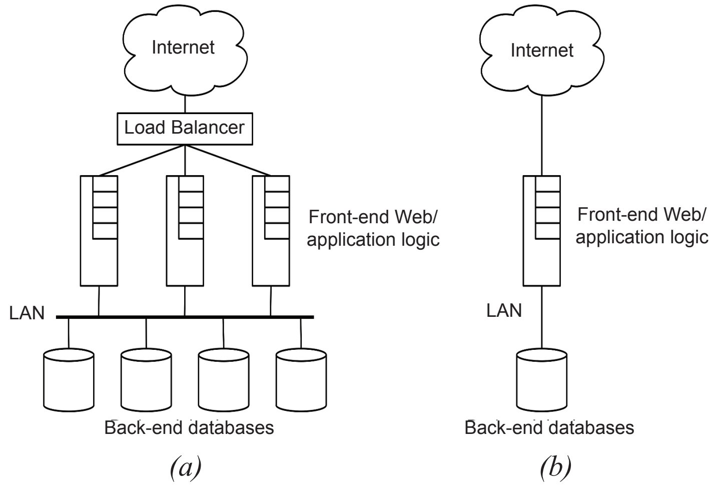
## DATABASE APPLICATIONS AND REPLICATION ISSUES

More and more companies and organizations are discovering the benefits of providing services over the Internet through portal applications. In order to provide a more profitable, responsive, and flexible user experience, these portal applications generate responses dynamically and provide a customizable experience for each user. That is, the content they provide to the user is generated on demand in response to user requests. These applications usually have two components: (1) front-end business logic that interacts with the user and provides the portal's look and feel and (2) a back-end database or databases containing the portal's true content (Candan & Li, 2002a; Li, Hsiung, Po, Hino, Candan, & Agrawal, 2004b; Vallamsetty, Kant, & Mohapatra, 2002). Based on user requests, the front-end logic queries the back-end databases, obtains needed content, and then uses that content to build a page (or other result), which is returned to the user. Figure 1(a) shows a diagram of a complex database-driven portal application while Figure 1(b) shows a simplified version containing the main components: the application logic and backend database. To emphasize the importance of the back-end database, such portal applications will be henceforth called *database-driven applications*, or DAs.

In order to meet user expectations and availability requirements, a DA must be able to return results quickly. Historically, this has been achieved by moving fragments of a DA's logic and its back-end database closer to end-users at the "Internet's edge" by caching raw data, caching results, or using fragmented page design (Choi & Luo, 2004; Datta, Dutta, Thomas, Vandermeer, & Ramamritham, 2004; Huang, Sebastine, & Abdelzaher, 2004; Li, Po, Hsiung, Candan, & Agrawal, 2003; Luo et al., 2002). These methods are generally compatible with dynamic Web page markup languages such as Java Server Pages and Edge Side Includes. However, such improvements can only provide so much relief. They may be attractive for some small or low-demand applications, but are not, in the authors' opinion, the ultimate solution for complex high-demand applications with large amounts of supporting data. Instead, full replication of the application and a significant part of its backend database will be needed in some cases.

Although it has many positive aspects, full replication has at least four challenges (in the authors' opinion) as outlined next.

*Figure 1. Multi-tiered portal application architectures. In (a) a complex arrangement of front-end servers is connected to a farm of database servers on the back-end. In (b) the arrangement is simplified to just the essentials.*


Internet
Load Balancer
Front-end Web/ application logic
LAN
Back-end databases
(a)
Internet
Front-end Web/ application logic
LAN
Back-end databases
(b)