Bulletin of the Technical Committee on

Data Engineering

December 1998 Vol. 21 No. 4

IEEE Computer Society

Letters

Special Issue on Data Replication

Conference and Journal Notices

The Bulletin of the Technical Committee on Data Engineering is published quarterly and is distributed to all TC members. Its scope includes the design, implementation, modelling, theory and application of database systems and their technology.

Letters, conference information, and news should be sent to the Editor-in-Chief. Papers for each issue are solicited by and should be sent to the Associate Editor responsible for the issue.

Opinions expressed in contributions are those of the authors and do not necessarily reflect the positions of the TC on Data Engineering, the IEEE Computer Society, or the authors' organizations.

Membership in the TC on Data Engineering (http: www. is open to all current members of the IEEE Computer Society who are interested in database systems.

The web page for the Data Engineering Bulletin is http://www.research.microsoft.com/research/db/debull. The web page for the TC on Data Engineering is http://www.ccs.neu.edu/groups/IEEE/tcde/index.html.

The Case for Non-transparent Replication: Examples from Bayou

Douglas B. Terry Karin Petersen Mike J. Spreitzer Marvin M. Theimer
Computer Science Laboratory
Xerox Palo Alto Research Center
Palo Alto, CA 94304 USA

Abstract

Applications that rely on replicated data have different requirements for how their data is managed. For example, some applications may require that updates propagate amongst replicas with tight time constraints, whereas other applications may be able to tolerate longer propagation delays. Some applications only require replicas to interoperate with a few centralized replicas for data synchronization purposes, while other applications need communication between arbitrary replicas. Similarly, the type of update conflicts caused by data replication varies amongst applications, and the mechanisms to resolve them differ as well.

The challenge faced by designers of replicated systems is providing the right interface to support cooperation between applications and their data managers. Application programmers do not want to be overburdened by having to deal with issues like propagating updates to replicas and ensuring eventual consistency, but at the same time they want the ability to set up appropriate replication schedules and to control how update conflicts are detected and resolved. The Bayou system was designed to mitigate this tension between overburdening and underempowering applications. This paper looks at two Bayou applications, a calendar manager and a mail reader, and illustrates ways in which they utilize Bayou's features to manage their data in an application-specific manner.

1 Introduction

A major challenge faced by designers of general-purpose replicated storage systems is providing application developers with some control over the replication process without burdening them with aspects of replication that are common to all applications. System models that present applications with ”one-copy equivalence” have been proposed because of their simplicity for the application developer [1, 3]. In particular, the goal of ”replication transparency” is to allow applications that are developed assuming a centralized file system or database to run unchanged on top of a strongly-consistent replicated storage system. Unfortunately, replicated systems guaranteeing strong consistency require substantial mechanisms for concurrency control and multisite atomic transactions, and hence are not suitable for all applications and all operating environments. To get improved levels of availability, scalability, and performance, especially in widely-distributed systems or those with imperfect network connectivity, many replicated storage systems have relaxed their consistency models. For instance, many systems have adopted an ”access-anywhere” model in which applications can read and update any available replica

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

and updates propagate between replicas in a lazy manner [2, 4, 7, 8, 9, 10, 12, 15]. Such models are inherently more difficult for application developers who must cope with varying degrees of consistency between replicas, design schedules and patterns for update propagation, and manage conflicting updates. The harsh reality is that applications must be involved in these difficult issues in order to maximize the benefits that they obtain from replication. The Bayou system developed at Xerox PARC is an example of a replicated storage system that was designed to strike a balance between application control and complexity.

This paper presents both the application-independent and application-tailorable features of Bayou along with the rationale for the division of responsibility between an application and its data managers. Examples drawn from a couple of Bayou applications are used throughout to illustrate how different applications utilize Bayou's features. The applications are a calendar manager and a mail reader. The Bayou Calendar Manager (BCM) stores meetings and other events for individual, group, and meeting-room calendars. A user's calendar may be replicated in any number of places, such as on his office workstation and on a laptop so that he can access it while travelling. Bayou's mail reader, called BXMH, is based on the EXMH mail reader [20]. BXMH receives a user's incoming electronic mail messages, provides facilities for reading messages, and permits the user to permanently store messages in various folders. The BXMH mail database managed by Bayou may be replicated on a number of sites for increased availability or ease of access. Each of these two applications interact with the Bayou system in different ways to manage their replicated data. They demonstrate the need for flexible application programmer interfaces (APIs) to replicated storage systems.

2 Application-independent Features of Bayou

For most replicated storage systems, the basic replication paradigm and associated consistency model are common to all applications supported by the system. While it is conceivable that a replicated storage manager could provide individual applications with a choice between strong and weak data consistency, this made little sense for Bayou. Bayou was designed for an environment with intermittent and variable network connectivity. In such a setting, mechanisms to support strong consistency would not be applicable. Therefore, Bayou's update-anywhere replication model and its reconciliation protocol, which guarantees eventual consistency, are central to the systems architecture. These fundamental design choices over which the application has little or no control are discussed in more detail in the following subsections. Additional application-independent mechanisms for replica creation and retirement are also briefly described. Features that are within an application's control, such as conflict management, are discussed in Section 3.

2.1 Update-anywhere replication model

Bayou manages, on behalf of its client applications, relational databases that can be fully replicated at any number of sites. Each application generally has its own database(s). Application programs, also called ”clients”, can read from and write to any single replica of a database. Once a replica accepts a write operation, this write is performed locally and propagated to all other replicas via Bayou's pair-wise reconciliation protocol discussed below. This ”update-anywhere” replication model, depicted in Figure 1, permits maximum availability since applications can continue to operate even if some replicas are unavailable due to machine failures or network partitions. Thus, it is particularly suitable for applications that operate in mobile computing environments or large internetworks. Because each read and write operation involves a single interaction between a client and a replica, the update-anywhere replication model is also easy to implement and provides good response times for operations.

This replication model was adopted for Bayou because of its flexibility in supporting a diversity of applications, usage patterns, and networking environments [6]. If replicas are intermittently connected, replicas are allowed to arbitrarily diverge until reconciliation is possible. If replicas are few and well-connected, the update-anywhere model is still a satisfactory choice since updates can propagate quickly under such circumstances and

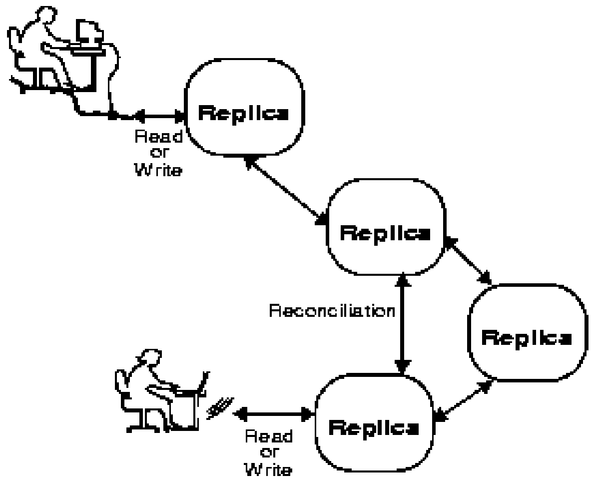

Figure 1: Bayou's update-anywhere replication model.

the replicas remain highly consistent. As described in section 3.1, applications can select reconciliation schedules that best fit their requirements for how much replicas are allowed to diverge.

Consider the example of a user, Alice, managing her personal calendar using BCM. Alice might keep a replica of her calendar on her office machine, one on her laptop, and also one on the office machine of her administrative assistant, Bob, so that her assistant has quick access to her calendar. Alice and Bob's office machines perform reconciliation with each other on a frequent basis so that any updates made to the calendar by either of them are seen by the other with little delay. However, when Alice is travelling, she may update the replica on her laptop while the laptop is disconnected. Any new meetings added by Alice are not readily available to Bob (and vice versa). From her remote destination, Alice occasionally connects to her (or Bob's) office machine via a dial-up modem to exchange recently added meetings, thereby updating their replicas of the shared calendar.

2.2 Reconciliation protocol and eventual consistency

Bayou's reconciliation protocol is the means by which a pair of replicas exchange updates or "writes" [16]. The protocol is incremental so that only writes that are unknown to the receiving replica are transmitted during reconciliation. It requires replicas to maintain an ordered log of the writes that they have accepted from an application client or received from another replica via reconciliation. Pair-wise reconciliation can guarantee that each write eventually propagates to each replica, perhaps by transmission through intermediate replicas, as long as there is an eventual path between a replica that accepts a write and all other replicas. The theory of epidemics indicates that, even if servers choose reconciliation partners randomly, writes will fully propagate with high probability [4]. Arbitrary, non-random, reconciliation schedules can be set up by applications if desired as discussed in section 3.1.

Bayou ensures "eventual consistency" which means that all replicas eventually receive all writes (assuming sufficient network connectivity and reasonable reconciliation schedules) and any two replicas that have received the same set of writes have identical databases. In other words, if applications stopped issuing writes to the database, all replicas would eventually converge to a mutually consistent state. Eventual consistency requires replicas to apply writes to their databases in the same order. Bayou replicas initially order "tentative" writes according to their accept timestamps, and later reorder these writes as necessary based on a global commit order

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.