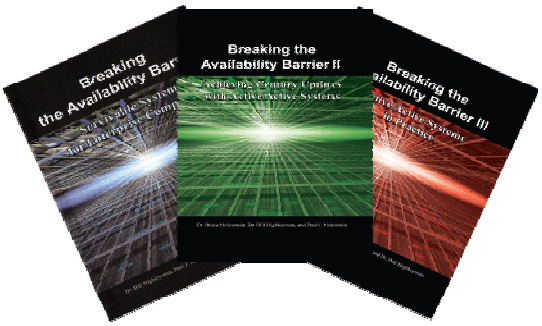
the Availability Digest

Breaking the Availability Barrier

May 2008



Breaking the Availability Barrier is a three-volume set[1] that focuses on active/active systems. According to the authors, "an active/active system is a network of independent processing nodes cooperating in a common application. Should a node fail, all that needs to be done is to switch over that node's users to a surviving node. Recovery is in subseconds to seconds."

Active/active systems don't just provide high availability; they provide continuous availability. They can provide availabilities in excess of six 9s and uptimes measured in centuries. This three-volume series covers in detail the logical and mathematical theory behind active/active systems, additional benefits that these configurations can provide, a review of current product technologies required for the implementation of these systems, and several case studies of active/active systems in successful production.

The authors attempt to make it possible to enter the volumes at any point and yet still understand the content. To that extent, most chapters start with a review of fundamentals applicable to the chapter topic. This leads to some redundancy but is intended to enhance the usability of the series by making most of the material stand-alone.

Throughout the series, a variety of rules are formulated that relate to continuous availability systems. These rules are summarized in the appendices of each volume.[2]

Breaking the Availability Barrier 1: Survivable Systems for Enterprise Computing

Part 1: Breaking the Four 9s Barrier

Volume 1 of this series begins with a general discussion of availability and reliability. A simple look at the analysis of redundant systems with multiple nodes and multiple spares is provided. The

[1] W. H. Highleyman, P. J. Holenstein, B. D. Holenstein, *Breaking the Availability Barrier: Survivable Systems for Enterprise Computing*, AuthorHouse; 2004.
B. D. Holenstein, W. H. Highleyman, P. J. Holenstein, *Breaking the Availability Barrier II: Achieving Century Uptimes with Active/Active Systems*, AuthorHouse; 2007.
P. J. Holenstein, B. D. Holenstein, W. H. Highleyman, *Breaking the Availability Barrier III: Active/Active Systems in Practice*, AuthorHouse; 2007.
[2] This summary was written by W. H. Highleyman, Managing Editor of the Availability Digest, who is one of the authors of the book series.

importance of repair time is introduced. The issue of repair time is expanded throughout the volumes as it is the cornerstone of active/active systems.

The concept of splitting a large monolithic system into a number of smaller systems all cooperating in a common application is then introduced. This gives a simplified view of active/active configurations. The need to replicate data between the databases of the cooperating systems is discussed. It is pointed out that organizing the processing nodes into pairs to support critical redundant processes can significantly improve the availability of the system.

Data replication engines are the heart of active/active systems. It is these engines that keep the databases that are distributed across the active/active network synchronized. Extensive discussions of asynchronous and synchronous data replication follow.

With respect to asynchronous replication, the problems of ping-ponging, data loss following a node failure, target database integrity and consistency, and data collisions are described. Data collisions, which occur when applications at two different nodes attempt to change the same data item at the same time, are perhaps the biggest problem when using asynchronous replication. Techniques for avoiding data collisions are discussed as well as techniques for detecting and resolving data collisions when collisions cannot be avoided. The issues involved in failing over and restoring full service following several failure scenarios are explored. There are many advantages to asynchronous replication, including the fact that it is noninvasive, scalable, and secure and imposes no performance penalty on the application.

Synchronous replication cures the data collision problem as well as the problem of data loss following a node failure. However, it extends the response time of applications since transactions must be committed across the network. Network deadlocks may be a problem should applications in different nodes attempt to lock the same data item at the same time. Two different approaches to synchronous replication are analyzed. One is network transactions, that are applicable to closely-located nodes when small transactions are the norm. However, network transactions may require significant application changes. The other approach is coordinated commits, in which data updates are propagated by asynchronous replication; and the updates are coordinated only at the end of a transaction for commit action. Coordinated commits do not require application modifications and are suitable for applications with high transaction rates, nodes separated by large distances, or for transactions that are large.

The question is asked, “Why do computers stop?” This leads to a discussion of the reasons for computer failures, including those caused by hardware, software, people, and the environment. It is pointed out that the predominant cause of failure is people, followed closely by software bugs. A further significant problem related to total system failover is failover faults. A failover fault occurs when a node fails and its backup fails to take over. This failure mode is extensively analyzed, and it is shown that a small probability of a failover fault can reduce availability drastically. The importance of recovery time is analyzed further, and the analysis concludes that recovery time is the most important parameter in achieving high availability.

In designing for high availability, it is important to set goals for recovery time and data loss following a failure. They are known as RTO (recovery time objective) and RPO (recovery point objective). A comparison of relative RTO and RPO is made for a variety of backup techniques, including tape backup, database replication to a cold standby, database replication to a hot standby, and various types of active/active configurations.

The compromises between availability, performance, and cost when building an active/active system are explored. Particular attention is paid to the configuration of the distributed database.

Part 2: Advanced Topics

Volume 1 concludes with detailed analyses of several of the topics covered in Part 1. These include data collision rates, referential integrity of the databases as they are synchronized by data replication, and failover faults. Extensive analyses using failure state diagrams are contained in the Appendices and support many of the intuitive relationships derived throughout the book.

Finally, a step-by-step procedure for implementing an active/active system is given.

Breaking the Availability Barrier II: Achieving Century Uptimes with Active/Active Systems

Volume II of this series delves into many additional topics of importance to active/active systems.

Part 1: Survivable Systems for Enterprise Computing

The book begins with a three-chapter summary of Volume 1. The concepts of availability and reliability are reviewed, and the strategy for achieving extreme availabilities with active/active systems are revisited.

An exhaustive review of the analyses behind active/active availability is provided. This review summarizes all of the analyses of Volume 1 and extends these analyses in several areas. It introduces the separate concepts of the four Rs - repair time, recovery time, restore time, and return to service time. The role that each of these plays in the achievement of high availability is clarified. The four Rs are the varied faces of MTR, which is usually known only as the mean time to repair.

Finally, active/active configurations are reviewed with emphasis on data-replication technologies. The various failure mechanisms that can impact an active/active system are reviewed as are the various compromises between availability, cost, and performance.

Part 2: Building and Managing Active/Active Systems

In this part, the various technical considerations involved in building a successful active/active system are discussed in some detail.

The discussion begins with an in-depth review of architectural and network topologies. The architectural topologies include application and database partitioning (which is useful for avoiding data collisions), asynchronous replication, synchronous replication, transaction replication, and hardware replication. While discussing asynchronous replication, significant detail on methods for detecting and resolving data collisions is provided. The introduction of the concepts of transaction replication and hardware replication extends earlier replication discussions. It is pointed out that hardware replication is not appropriate for active/active systems because it does not provide for a consistent replicated database that can be used by other applications on the target system while replication is occurring.

The use of active/active systems to provide *fair* data access to all users is discussed. Active/active systems can be configured to equalize the response time among all users for data query and update functions.

The discussion of network topologies describes various network configurations that can be used to build active/active systems along with the advantages and disadvantages of each configuration. These configurations include hot standbys (sizzling-hot takeover), bidirectional active/active networks, route-thru configurations, ring networks, hierarchical configurations, and fully and partially connected networks.

Following this review, the implementation of redundant reliable networks is presented. The discussion covers both local area networks and wide area networks. It points out that the Internet Protocol (IP) has become the de facto standard for networked communications, in part because of its recovery capabilities in the face of network outages or bottlenecks. The detection of network faults via heartbeats, missing responses, TCP detection mechanisms, and path monitoring are explored. Network fault recovery can be via a variety of mechanisms including alternate routing or the failover to backup networks. The use of virtual IP addresses can simplify recovery for clients significantly.

One decision that the implementers must make is whether to use manual or automatic recovery. Automatic recovery can be much faster, but network problems can be very subtle; and consequently manual recovery may be more effective. Also discussed are the concerns following a network failure, concerns such as transaction loss, session loss, and connection loss.

Next, the issues associated with distributed databases are reviewed. Issues with distributed databases include replication engine latency (the time from when an update is made to a source database to the time that it appears in the target database), application latency (the increase in transaction response times due to synchronous replication), data loss following a node failure, and data collisions. One major issue is called the *database of record*. If there are multiple copies of the database in the application network, which one should be considered the *single version of truth* in the event of inconsistencies between the database copies? This topic is explored in some detail.

With regard to remote access of databases, there are several issues which are discussed. A node may have to access data on a remote node either because of the system configuration or because of a database failure. The failover to a backup database and the recovery of a failed database are discussed.

The causes of nodal failures are explored. The detection of a node failure and the failover of that node's users to a surviving node are explained. Several techniques for quickly moving users from a failed node to a surviving node are covered. They include the use of virtual IP addresses, router redirection, gratuitous ARP, and manual switchover. Other issues to be considered are the shedding of low-priority load if a surviving node must accept additional load from the users of a failed node, split-brain operation (nodes continuing processing without the ability to coordinate with each other), and tug-of-war, during which two different nodes each think that the other has failed and try simultaneously to take over the application load of the other node.

One major advantage of active/active systems is that they can eliminate planned downtime. This is supported by a technique known as Zero Downtime Migration (ZDM). ZDM starts with removing a node from the application network and transferring that node's users to other nodes in the network. The node that has been removed can then be upgraded, whether it be a hardware upgrade, a new operating system version, a new database version, or an upgraded application. The node is then returned to service, resynchronized to the current production database, and allotted users. The upgrade is rolled through the other nodes in a similar manner. The requirement for an online copy facility that can resynchronize a database while the source database is active is discussed.

It is not necessarily a trivial operation to convert applications that are not currently running in an active/active mode to run in an active/active network. Several of the considerations that may require application modification are discussed.

Finally, the considerations in calculating the total cost of ownership (TCO) of an active/active system are explained. Definitions of net present value (NPV), internal rate of return (IRR), and return on investment (ROI) set the stage for this discussion. Various initial and recurring cost factors are analyzed. Initial cost factors include the initial system costs of hardware acquisition, storage configurations, and one-time software license fees. Recurring costs include hardware and software maintenance, licensing, networks, personnel, facilities, and the cost of downtime. The

cost factors that an active/active system can reduce are business insurance and the all-important cost of downtime.

Breaking the Availability Barrier III: Active/Active Systems in Practice

Volume III of the series is actually an extension of Volume II. Volumes II and III had originally been intended to be one book, but it simply became too big and had to be split into separate volumes.

Part 3: Infrastructure Case Study

Active/active systems depend upon data-replication engines to keep the various database copies synchronized. In this part are reviewed some commercially available products that are examples of data replication engines and online copy utilities needed to implement active/active systems. The authors' are most familiar with data replication and active/active products from Gravic, Inc., and the products reviewed are those from Gravic.

Part 3 starts with a performance model of a data-replication engine. To minimize data loss following a node failure and to minimize data collisions, it is important that the replication latency of the data-replication engine be as small as possible. The performance model, which is detailed in an Appendix, shows that the replication latency depends only upon a small set of parameters. These include the number of disk queuing points within the engine, whether polling of queues is needed or the engine is event-driven, communication buffering, and communication channel latency (the time to send a signal over the communication channel). If the replication engine is multithreaded, the time that it takes to reserialize the transactions is also a factor. In addition, the application latency imposed by a synchronous replication engine is also analyzed.

The Shadowbase replication engine from Gravic is described in detail. It provides bidirectional replication services between homogeneous and heterogeneous platforms and databases. Full on-the-fly transformation and mapping facilities are available as standard features or through user exits.

The online copy facility that is described is Gravic's SOLV (Shadowbase Online Loading, Validation, and Verification utility). SOLV can copy an active source database to a target database without having to quiesce either the source or target database. The portion of the target database that has been copied is consistent and up-to-date and can be used by applications running on the target system. SOLV also provides online validation and verification of two databases. Verification reports any differences in the databases. Validation brings one database into correspondence with the other so as to guarantee that they are identical.

The use of Shadowbase and SOLV for zero downtime migration is described.

Part 4: Active/Active Systems at Work

Active/active systems bring many additional benefits to an application than just the continuous availability features described above. Continuous availability features include very fast recovery time (subseconds to seconds), the elimination of decision time from the recovery process, the elimination of planned downtime, and disaster tolerance. Other benefits include improved data locality, the efficient use of all available capacity, risk-free failover testing, application scaling, lights-out operation, and load balancing.

Several case studies of successful active/active systems currently in production are given. They include systems running applications in the fields of financial services, telecommunications, manufacturing, insurance, travel, and gaming.

Finally, a series of related technologies are described. They include the system structures for HP NonStop servers and IBM Parallel Sysplex systems, the Real-Time Enterprise technology that

allows companies to react immediately to events, grid computing, and virtual tape. A variety of regulatory requirements are summarized.

The volume concludes with a critique by an industry analyst of active/active systems.

Summary

The authors of this series have had extensive experience implementing active/active systems, having been involved in the implementation of these systems for over fifteen years. They hold several patents on active/active technology and speak and write frequently on the subject. Their books are available from several sources such as Amazon.com (http://www.amazon.com/s/ref=nb_ss_b?url=search-alias%3Dstripbooks&field-keywords=Breaking+the+Availability+Barrier&x=13&y=15).

Compared to the mature cluster technology that is so well known today, active/active systems are the newcomers. On the one hand, clusters have been around for decades. They are supported by most major manufacturers and a wealth of third-party products. However, active/active systems can provide an order-of-magnitude greater availability than clusters and can reduce or even eliminate the effects of systems failures on users. Product support for active/active technology is now available off-the-shelf, and many successful implementations are installed and have been running for years without a system failure.

Clusters provide high availability. Active/active systems provide continuous availability.

By the way, it is rumored that a Volume IV is in the works …