"Achieving Century Uptimes"
An Informational Series on Enterprise Computing

As Seen in *The Connection*, A Connect Publication
December 2006 – Present

About the Authors:

Dr. Bill Highleyman, Paul J. Holenstein, and Dr. Bruce Holenstein, have a combined experience of over 90 years in the implementation of fault-tolerant, highly available computing systems. This experience ranges from the early days of custom redundant systems to today's fault-tolerant offerings from HP (NonStop) and Stratus.

Gravic, Inc.
Shadowbase Products Group
17 General Warren Blvd.
Malvern, PA 19355
610-647-6250
www.ShadowbaseSoftware.com

Achieving Century Uptimes
Part 12: The Rules of Availability III
September/October 2008

Dr. Bill Highleyman
Dr. Bruce Holenstein
Paul J. Holenstein

We conclude in this article a review of our "Rules of Availability," as published in our series of books entitled *Breaking the Availability Barrier*.[1] We have chosen those rules that are particularly applicable as *best practices* to achieve continuous availability with redundant systems and with a focus on active/active systems.

The Importance of Recovery Time

There has been a paradigm shift in the approach to continuous availability. Almost four decades of development have resulted in fault-tolerant systems that today are so reliable that they hardly ever fail due to hardware or operating system faults. Any further efforts to improve these reliabilities are overshadowed by other faults such as application errors and operator errors. As a result, we have seen a shift from trying to prevent failures through fault-tolerant system design to accepting that failures will occur and to being able to recover from these failures so quickly that no one notices. In other words, *Let it fail, but fix it fast.*

Rule 41: *Active/active systems can provide the availability of a primary/standby pair with less equipment and less cost.*

When you back up your data-processing system with a passive/standby system, you must purchase 200% capacity – 100% for your primary system and an equivalent 100% for your backup system. But half of this capacity goes unused. However, the probability of a total system outage is extremely small since it requires the failure of both nodes.

In contrast, an active/active system can be configured with multiple smaller nodes. For instance, consider a five-node active/active system with each node carrying 25% of the load. You only have to purchase 125% of capacity rather than 200%. It still takes a two-node failure to drop below 100% capacity, but in this case 75% of capacity is still available (rather than no capacity in the primary/standby case). Furthermore, the time required to return service to the affected users is typically measured in seconds rather than in hours.

---

[1] *Breaking the Availability Barrier: Survivable Systems for Enterprise Computing*, AuthorHouse; 2004.
Breaking the Availability Barrier II: Achieving Century Uptimes with Active/Active Systems, AuthorHouse; 2007.
Breaking the Availability Barrier III: Active/Active Systems in Practice, AuthorHouse; 2007.
These rules may also be found at http://www.gravic.com/breaking_the_availability_barrier_rules.html.

Rule 42: *In a system with s spares, reducing subsystem mtr by a factor of k will reduce system MTR by a factor of k and will increase system MTBF by a factor of k^s, thus increasing system reliability by a factor of k^{s+1}.*

This rule states the importance of being able to repair a node in a redundant system quickly. For instance, consider a dual-node system with one spare (the normal case). Assume that a dual-node failure will take down the system once every five years (the system MTBF). Furthermore, assume that a node has a mean time to repair (nodal mtr) of four hours. Should both nodes fail, it will take an average of four hours to return the system to service (assuming exponential repair times and only one repair person who works only on one node at a time). This is the system MTR.

If the nodal mtr can be cut in half to two hours, the system MTR will be cut in half to two hours. Because the window during which the second node can fail (the time that one node is down) is now halved, the system MTBF will be doubled to ten years. The probability of a failure of the system is MTR/MTBF. Therefore, the probability of a failure has been reduced by a factor of four. The reliability of the new system has been increased by a factor of four by cutting the nodal repair time in half.

Rule 43: *If s+1 subsystems fail and are being repaired simultaneously, and if the return to service of any one of these subsystems will return the system to service, the system MTR is mtr/(s+1).*

In a system with *s* spares, it takes the failure of *s*+1 nodes to cause the system to fail. The repair of any one of these nodes will return the system to service. If there are *s*+1 repair people, one for each spare node, then on the average there will be *s*+1 repairs made during the time that it takes to repair one node. Assuming that repair times are random with each nodal repair taking an average time of mtr, then the rate of repairs is (*s*+1)/mtr; and the average system repair time, MTR (which equals the time to repair the first node), is mtr/(*s*+1).

For instance, if there is one spare in the system, the system MTR will be cut in half if two repair people are available rather than just one.

Rule 45: *If you break a monolithic system into k smaller nodes with no spares, the system will be more reliable than the original monolithic system provided that each node is more than k times as reliable as the monolithic system.*

In many systems such as multiprocessor systems, the smaller the system, the more reliable it is since there are fewer ways for it to fail (less parts, hence more reliability). For instance, consider a large single-spared system with 8 subsystems. In this case, there are 8x7/2 = 28 ways in which two nodes can fail (the number of failure modes) and take down the system.

Now let us split this system into two four-node systems. Each node only has six failure modes (there are six ways that two nodes can fail in a four-node system). However, there are two nodes in the system; and the failure of any node is assumed to take down the system. Thus, there

are a total of 2x6=12 failure modes in the split system. The dual-node system is more than twice as reliable as the monolithic system.

Rule 46: *Don't underestimate your failover time. It may well be the most important factor in perceived availability.*

If the nodes in your distributed system are so reliable that a dual-node failure is highly unlikely, then user downtime reduces to the time that it takes to recover services for the users who have been affected by an outage. This is the failover time, and it becomes the predominant factor in the reliability equation. In the limit, if the failover time is so short that the user has not noticed the outage, then in effect there has been no outage.

Rule 47: *A small probability of a failover fault may cause a disproportionate decrease in system availability. Moving from active/backup to active/hot-standby or active/active with frequent testing can significantly reduce the probability of failover faults.*

Rule 46 assumes that the failover will work. However, there are many reasons that a failover will fail. We call this a *failover fault*. Most failover faults are related to inadequate testing of the failover procedures. In the example given in our book, a 1% chance of a failover fault (that is, a failover works 99 times out of a 100 – wouldn't that be nice) results in a 500% decrease in availability!

Active/active systems substantially eliminate the chance of a failover fault because it is known that the backup system is operational. After all, it is actively processing transactions at the time that another node fails.

Rule 48: *Pick your node locations in an active/active system carefully to minimize the chance that environmental hazards will outweigh the availability of the nodes.*

What is the advantage of using highly reliable systems in a data center if there is a significant probability that the data center may be taken down by outside events? Try to locate data centers outside of hurricane and tornado zones, flood plains, earthquake fault lines, and so on.

Rule 49: *Replication engines that violate referential integrity on the target database are rarely, if ever, suitable for active/active implementations.*

If a data-replication engine does not ensure referential integrity of the target database, there may be a significant time required to bring the target database into a state of full consistency following a failure of the node that it is backing up. This can lead to very long failover times.

Active/active systems, of course, require a data-replication engine that guarantees referential integrity and transactional consistency at the target database since the target database is being actively used.

Rule 50: (Corollary to Rule 49) - *Hardware replication is not suitable for active/active applications because it does not provide referential integrity.*

Database systems in which physical block replication (such as replicating cache block changes) are replicated by the hardware storage controllers do not provide referential integrity since there is no sequential control over the replicated blocks. Indices may be replicated that have no associated data rows, children may have no parents, and so forth.

Rule 53: *ZDM (zero downtime migration) eliminates planned application downtime, therefore improving application availability.*

A real killer of availability is planned system downtime for upgrades, migrations, backups, batch runs, and so forth. This overrides failover as a consideration since users can be out of service for hours. Active/active systems eliminate planned downtime because nodes can be taken out of service one at a time, upgraded or used in whatever way is necessary, and then returned to service. While the node is out of service, users are routed to other operating nodes.

Total Cost of Ownership

Rule 54: (Standish Law) - *In order to calculate a meaningful predicted Total Cost of Ownership, one must first properly size the system.*

Although this seems obvious, a major obstacle to proper system sizing is often the system vendor. Each vendor will maximize the capability of his offerings (often called *puffery*), which may lead to an undersizing of the system. In addition, reliability claims may be excessive and thus result in unrealistically low estimates of the cost of downtime.

It is important to fully understand the performance and availability parameters of any system that you are considering so that initial and operating cost estimates can be accurate.

Rule 56: *To minimize insurance costs, reduce threat by increasing mtbee (the mean time between endorsed events), reduce vulnerability by using appropriate countermeasures, and minimize the value of an event by minimizing MTR. An active/active system is an effective countermeasure that reduces both vulnerability and event value.*

The use of active/active systems often can significantly reduce business insurance costs. This is because the extreme availability of an active/active system ensures against business disruptions. Extending an active/active system so that it is geographically dispersed further protects against business interruptions caused by site and area disasters.

Rule 57: (Corollary to Rule 39) - *The use of fault-tolerant nodes in an active/active system can reduce TCO by reducing the number of nodes required, the facilities' costs, the licensing costs, and the personnel costs.*

Rule 39 states that the use of active/active systems will generally require fewer nodes if fault-tolerant nodes are used rather than industry standard high-availability nodes to achieve a specified system availability. In most cases, at least one less node of the same capacity will be required when fault-tolerant nodes are used due to reduced sparing requirements.

Performance

Rule 58: (Latency Rule) - *Replication engine latency is largely governed by disk-queuing and communication delays, not by processing times.*

When analyzing the performance of data-replication engines, it is found that processing time is negligible compared to other delays. Predominant among these other delays are the access time and polling delays associated with disk storage and communication delays.

Disk-storage delays can be minimized by reducing the number of disk-queuing points in the replication engine. Communication delays (aside from channel latency) are primarily caused by the time to buffer large blocks of data for efficient channel utilization. In this case, replication latency can be traded for efficient channel utilization.

Rule 61: *Active/active systems made up of fault-tolerant nodes will benefit from lights-out operations. Active/active systems using high-availability (but not fault-tolerant) nodes may have, in fact, poorer availability in a lights-out operation than the individual nodes.*

This is especially true if there are multiple lights-out nodes. Let us assume that the repair time for a lights-out node is 20 hours rather than two hours for a staffed node. In effect, a lights-out node will have one less 9 of availability than a staffed node.

Consider a five-node active/active system in which four nodes are lights-out nodes monitored by one staffed node. The system will survive the loss of any two nodes. If four-9s, fault-tolerant servers are used, the resulting availability is a little less than six 9s. However, if three-9s, high-availability servers are used, the net availability is a little over three 9s – about the same as if a single node were used.

Rule 62: (Paul's Law) - *If synchronous replication is used, the Applier does not have to enforce the order of commits so long as the source referential integrity is at least as strong as that at the target.*

As opposed to asynchronous replication, a synchronous replication engine guarantees that the order of transactions applied to the target database is the same as that at the source database. Therefore, if the source database is ensuring referential integrity, the target database will ensure the same level of referential integrity.

The one case where this might be a problem is if the target database enforces a higher level of referential integrity than the source database. In this case, referential integrity violations that were not enforced by the source database could occur at the target database.

Rule 63: *The difference in application latency between coordinated commits and dual writes is most apparent when fast response times are required.*

The difference in application latency between dual writes (synchronizing at the write level) and coordinated commits (synchronizing at the transaction level) is measured in the tens or hundreds of milliseconds depending upon the distance between the two nodes. Dual writes are faster for short distances and short transactions, and coordinated commits are faster for long distances and/or long transactions.

If users are expecting subsecond response times, the choice of the synchronization technique may be important. However, if users are expecting response times measured in seconds (for instance, for web applications), the difference between these techniques so far as the user experience is concerned may not be significant.

Application Active/Active-Ready

Rule 64*:* (Werner's Law) - *When trying to distribute an application over multiple nodes, identify each global resource and local context the application uses; and carefully consider the consequences of this in a multinode approach. Identify each application decision which is made on system or database state information (as opposed to database contents information), and make sure this is compatible with your multinode approach.*

Although in principle an application may be made active/active by running it in multiple nodes whose databases are synchronized by data replication, not all applications may be active/active-ready. For instance, if sequential or random numbers are generated, multiple nodes may generate identical numbers. Important memory-based context may not be replicated. Duplicate mini-batches may be run at each of the nodes. Distributed deadlocks may occur.

Therefore, an application must be thoroughly analyzed; and any required modifications must be made before attempting to run it in an active/active mode.

In Memoriam

The database pioneer Jim Gray, lost at sea on January 28, 2007, contributed greatly to the technology that today allows active/active systems to be built. After reviewing the first book in this series, he commented, "I loved the laws!" We hope that you have enjoyed them and have learned from them as well.