

## **Choosing a Business Continuity Solution Part 1 – Availability Fundamentals**

July 2011

Business continuity encompasses those activities that an enterprise performs to maintain consistency and recoverability of its operations and services. The availability of application services provided by an enterprise's IT infrastructure is only one of many facets of business continuity, albeit an extremely important one. Application availability depends upon the ability of IT services to survive any fault, whether it is a server failure, a network fault, or a data-center disaster. An enabling technology for achieving high availability and even continuous availability for application services is data replication. Selecting the right data-replication technology to achieve your business continuity goals is the focus of this four-part series.<sup>1</sup>

Availability doesn't come for free. Every organization has a variety of IT applications, the importance of which ranges from noncritical to mission-critical. Some applications can be down for hours or even for days with little impact on operations. Downtime for other applications may be quite costly but survivable. A handful of applications cannot be down at all without loss of critical services, loss of life, or without otherwise wreaking major havoc on operations.

Fortunately, a range of data-processing architectures exists for meeting every application-availability need. However, a common characteristic of such architectures is that the more availability they provide, the more costly they become and the more complex they are to implement and to manage.

This series of articles provides management with an understanding of data-replication technologies. Management can then make informed decisions concerning the availability approach to be applied to each application for maximum return on the company's investment.

In this first part, we review many of the fundamental concepts that help us define availability.

### **Availability Fundamentals**

Congratulations! If you are reading this article, you have taken an important step toward solving your availability needs. We begin with the fundamental requirements to be met by any system for which availability is an issue.

---

<sup>1</sup> This series of articles is a reprint of a Gravic, Inc., white paper and is published with the permission of Gravic. See the Gravic web site for other white papers at [www.gravic.com/shadowbase/whitepapers](http://www.gravic.com/shadowbase/whitepapers).

## ***Redundancy***

Single points of failure should be eliminated. This means that every component that is necessary to support an application should be backed up by an equivalent (not necessarily identical) component. Redundancy applies to processors, data storage, networks, sites, power, cooling, and others.

In some cases, a company may elect not to back up certain components that are considered highly unlikely to fail.<sup>2</sup> In such cases, the company is willing to take the consequences of the (presumably) infrequent failure of those components.

## ***Isolation***

Redundant components should be isolated so that the failure of one does not affect its backup. For instance, depending upon two power feeds into a data center from the same power grid violates isolation since if one fails due to a grid fault, the other fails. Likewise, dual networks from the same communication carrier violate isolation.

Another violation of isolation is inherent in those architectures that require a backup system to be identical to the primary system, even down to the hardware, the software versions, and the database schema. One problem associated with the requirement for identical facilities is that a bug in one system may simply reoccur in the other system following a failover. Another is that it is often difficult for system administrators to know if changes made in one system do, in fact, make it to the other system. These problems often occur in active/passive system configurations and in clusters.

Truly isolated redundancy implies that the backup facility does not have to be identical to the primary facility. What is necessary to satisfy the isolation criterion is that a problem or a configuration change affecting one system does not impact the other system.

## ***Dispersion***

Redundant components must not be collocated. They must be geographically dispersed so that some single event cannot disable both of the redundant components.

An important single point of failure is the data center. Reports occur regularly of entire data centers going down for hours or days. Disasters such as fires, floods, tornados, and earthquakes as well as more subtle events such as employee malfeasance, hackers, and even law-enforcement confiscation of data-center equipment can cause data-center outages.

The only solution to this problem is to have two or more data centers located sufficiently far from each other so that a disaster at one site does not affect other sites. Fault-tolerant architectures such as a single HP NonStop system, clusters, and virtual machines are highly redundant but are inherently not dispersible. Though they form a solid basis for a highly-available or continuously-available data center, true high availability means that they must be replicated in a distant data center.

The distance by which data centers must be separated depends upon the threats. In Europe, for instance, where there is no expectation of earthquakes or hurricanes, data-center separations of tens or hundreds of kilometers are often considered adequate to protect against local disasters

---

<sup>2</sup> "Highly unlikely to fail" does not mean failure-free. In September, 2006, Valerie Wilson won the million-dollar New York State Lottery for the second time. Her chances of doing this were one in 3.6 trillion. That is more than twelve 9s. But it happened. Beware of letting a high number of 9s lull you into a false sense of security. The question is not, "Can it fail?" The question is "When will it fail?" More importantly, what are you willing to do to prepare for such an event? After all, that highly unlikely event may just happen tomorrow.

such as fires, floods, and power outages. In other, more disaster-prone areas, hundreds or thousands of miles are frequently required.

Additionally, local regulations may play a role in dictating distances. For instance, European countries generally dictate their own distance requirements within the country for critical applications. In the United States, federal regulations call for a minimum distance measured in the hundreds of miles for critical financial applications.

### **Failover**

A redundant component is useless if it cannot readily assume the duties of its failed counterpart. This requires multiple capabilities to be part of the architecture:

- *Fault detection* to rapidly identify faults.
- *Failover decision* to determine whether it is better to try to restore the failed component (for instance, to reboot a processor) or to fail over to its backup.
- *Failover action* to perform the failover if this is the desired action.
- *Verification* to validate that the backup component is providing the required service to the system.

To the extent that these capabilities are automated, failover will be faster. It also typically will be more reliable and less error-prone. However, in many cases, one or more capabilities are deemed too complex for automation and are left to the system operators and their management. For these cases, established procedures, thorough documentation, and periodic operator practice are important.

### **Testing**

Perhaps the availability requirement that is most violated is periodic testing of the failover plan. In many architectures, failover testing requires stopping processing and failing over to the backup, a course of action that might take down user access to the application for hours. Added to this is the possibility that the failover will fail, further increasing the outage. For this reason, many companies don't test their failover plans or only do so rarely or in part. When a real failure occurs, and they must fail over, all they can do is to hope for the best.

As will be seen, some architectures lend themselves to continuous backup-system testing without taking the system offline so that the operations staff always knows that the backup system is working properly.

## **The Components of Availability**

Many factors influence the availability of data-processing systems. Factors range from planned outages to unplanned outages. In this white paper, we will discuss the use of data-replication techniques for reducing or eliminating both planned and unplanned outages.

### **Planned Outages**

A processing system or a node in a redundant processing system may have to be taken down periodically for planned maintenance. If the system is not redundant, the users are down during this planned maintenance interval. However, if the system is redundant, another node in the system can continue to provide processing services.<sup>3</sup>

---

<sup>3</sup> [Using Shadowbase to Eliminate Planned Downtime via Zero Downtime Migrations, A Gravic White Paper.](#)

Often, for nonredundant systems, there is a maintenance window during which the system is idle and during which such upgrades can be performed. Perhaps the window is at night or over the weekend. But will the maintenance activities be completed on time? Will the system come up properly at the end of the maintenance window?

Planned outages are often necessary for hardware upgrades. Also, many software upgrades require that the system be taken down. They include upgrades to the operating system, applications, and database-management systems. Certain database operations such as rebalancing indices, modifying the database schema, or making consistent tape backups may also have to be performed on a quiescent system.

Typically, a data-center site move necessitates that all systems be taken down.

### ***Unplanned Outages***

An unplanned outage requires either an immediate recovery of the failed component or failover to a redundant component, if there is one. Any number of faults can cause an unplanned outage, including hardware failures, software bugs, operator errors, environmental faults such as power or air conditioning, employee malfeasance, hackers, denial-of-service attacks, and even data-center destruction due to a disaster of some sort.

There are two major availability concerns when an unplanned outage occurs – the availability of the processing infrastructure and the availability of the application data. Having all of the servers up and running does no good unless they have valid data upon which to operate. We look next at processing availability and data availability.

## **Application-Services Availability and the Recovery Time Objective (RTO)**

The availability of the processing infrastructure – servers, networks, power, etc. – varies widely depending upon the redundant architecture used. We will explore these architectures later. First, let us look at how we characterize the availability of processing services.

### ***Uptime***

The typical way to measure availability is to predict (or measure) the percentage of time that the system is up – its *uptime*. Uptime is often reported in “nines” (9s).<sup>4</sup> For instance, if a system is up 99.9% of the time, its uptime (thus, its availability) is quoted as “three 9s.” The relationship between 9s and downtime is shown in Table 1.

<b>Uptime</b>	<b>Downtime/year</b>
one 9	876 hours
two 9s	88 hours
three 9s	9 hours
four 9s	50 minutes
five 9s	5 minutes
six 9s	30 seconds

**Downtime and 9s**  
**Table 1**

---

<sup>4</sup> W. H. Highleyman, P. J. Holenstein, B. D. Holenstein, Chapter 1, [The 9s Game](#), *Breaking the Availability Barrier: Survivable Systems for Enterprise Computing*, AuthorHouse; 2004.

Keep in mind that these are averages. Four 9s, for instance, does not mean that you will experience a one-hour outage each year. You will achieve four 9s if you have a four-hour outage every four years or a one-day outage every twenty-four years. Can your business survive that?

It is up to you to decide the criticality of your various applications and the downtime that you are willing to tolerate for each.<sup>5</sup> In this series, we concentrate on techniques for achieving five 9s of availability and beyond.

### ***Disaster Recovery versus Disaster Tolerance***

*Disaster recovery* is the ability to bring up backup facilities to carry on the business following the loss of data processing. This may require recovering the database, loading applications, testing the backup site before IT operations can resume, and finally switching the network and the users over to the recovered system. Disaster recovery typically takes hours or days.

*Disaster tolerance*, on the other hand, is the ability to continue operations in the event of a disaster without users noticing that there has been an outage or at least without denying application services to the users for very long. How long is “very long”? This depends upon your application requirements, as discussed below.

In this series, we focus on achieving disaster tolerance for your critical applications.

### ***High Availability***

Typically in the industry today, *high availability* is seen as an availability of five 9s or greater (less than five minutes of downtime per year). Again, keep in mind that this is an average. It is achieved if a system fails five minutes once per year or one hour every twelve years.

### ***Continuous Availability***

If an application requires an availability of six 9s or beyond, the measure of 9s is no longer very helpful. Measuring seconds of downtime per year is simply not practical. Rather, what is important is failover time and failover reliability. If the system can fail over from a hardware or software fault or even a data-center disaster in subseconds or seconds, and if it is known that the failover will succeed, then true disaster tolerance as defined above is achieved. This is *continuous availability*.

### ***Recovery Time Objective (RTO)***

The failover time that you can tolerate for an application is defined as the *Recovery Time Objective*, or *RTO*.<sup>6</sup> We will use this measure extensively in the following discussions.

Data-replication technology is a key enabler to achieve either high availability or continuous availability by minimizing RTO.

## **Data Availability and the Recovery Point Objective (RPO)**

The previous section dealt with system availability. But an available system with full processing capacity is of no use if it does not have correct and current data to process. In some cases, access to a complete history of activity is also required.

---

<sup>5</sup> This is obviously a business decision, as the amount of availability that the business requires will drive the architectures to consider and therefore the cost and complexity of the system to be implemented.

<sup>6</sup> W. H. Highleyman, P. J. Holenstein, B. D. Holenstein, Chapter 6, *RPO and RTO*, *Breaking the Availability Barrier: Survivable Systems for Enterprise Computing*, AuthorHouse; 2004.

There are many reasons why important data may be lost, from system failures to data-center failures. Data is typically protected by backing it up. Should there be a failure of some sort, the data can be restored from its last backup. Any data updates that have occurred since the last backup are lost.

Each application has a different importance to the company, and the importance of the application-generated data will vary. In some cases, the data has little value. In other cases, the loss of a few minutes of data, while painful, may be acceptable. For mission-critical applications, zero data loss may be required.

### **High Data Loss**

Perhaps the data associated with an application has little value. Hours of data can be lost with no serious impact to the company. Data used only for statistical analysis, such as counting web clicks, is an example of this type of data.

### **Little Data Loss**

For many applications, data is very important to the company. Though its loss will not imperil the company, the cost of losing this data might be very high. In some cases, it might be possible to recreate the lost data via manual data entry once the system is restored so long as not more than a few minutes of data are lost. Examples of this kind of data include ATM transactions and cell-phone call data records.

### **No Data Loss**

Some applications represent the core of the business, and no loss is tolerable. Any lost data may be unrecoverable, and such loss could seriously impact the fundamental operations of the business. An example is the trading data for a stock exchange. If trade data is lost and cannot be reconstructed, there is no basis for establishing the price of the various instruments traded on the exchange. In these cases, data loss cannot be tolerated. Other examples are electronic funds transfers (EFT) and health-care records.

### **Recovery Point Objective (RPO)**

The amount of lost data that an application can tolerate is its *Recovery Point Objective*, or *RPO*.<sup>7</sup> The data-replication techniques that this series discusses can reduce RPO to seconds and even to zero if required.

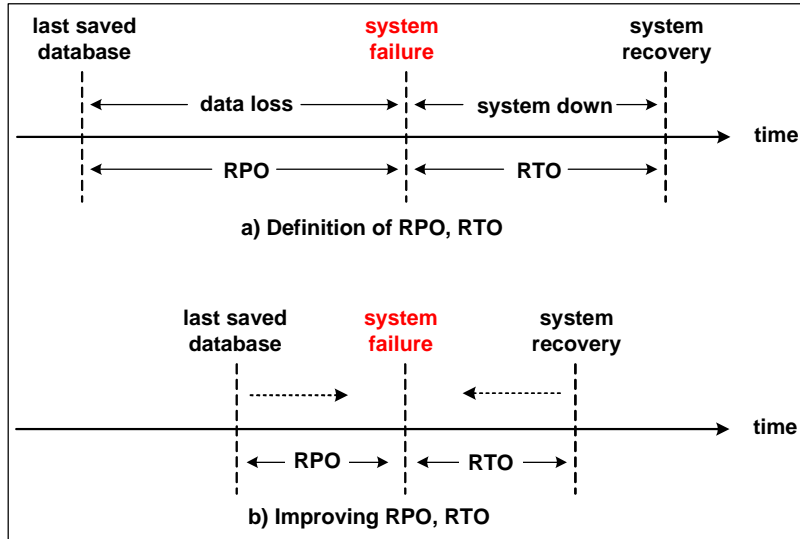
## **Specifying Availability – RTO and RPO**

In our discussions above, we introduced the concepts of RTO (Recovery Time Objective) and RPO (Recovery Point Objective) for an application. RTO is expressed as time and is the amount of time that the company can afford to operate without the data-processing services of an application. RPO is also expressed as time and is the amount of application data loss that the company can tolerate due to an unplanned outage.<sup>8</sup>

---

<sup>7</sup> W. H. Highleyman, P. J. Holenstein, B. D. Holenstein, Chapter 6, *RPO and RTO*, *Breaking the Availability Barrier: Survivable Systems for Enterprise Computing*, AuthorHouse; 2004.

<sup>8</sup> Expressing data loss in terms of time may seem counterintuitive. In real systems, however, it is a practical method to use. It is the time from the generation of data to the time that it is safe-stored on the target system that is really being measured. This can be converted directly to lost data by using the transaction rate. If the RPO is one minute, and if transactions are being generated at a rate of 1,000 transactions per minute, this is equivalent to specifying that no more than 1,000 transactions can be lost.



**Figure 1: The RPO/RTO Relationship**

In Figure 1a, we show the relationship between RPO and RTO. Backups might be made by quiescing the system and by dumping the database to magnetic tape, by executing online dumps to disk, by using audit-trail rollups, or by employing a number of more immediate techniques such as data replication, as discussed later. Should the primary system fail, the last database backup can be loaded onto a backup system; and processing can continue. The amount of application data lost is that new data generated from the time of the last database backup or replication point to the time of failure. This is the maximum amount of data that can be lost and yet still achieve the desired RPO goal.

Techniques for moving the left vertical bar to the right, closer and closer to the point of system failure, thereby reducing or eliminating data loss, as shown in Figure 1b, is one focus of this series.

Following a failure, steps are taken to recover processing. This may entail repairing the downed system, or it may require switching over to a backup system. The time from the point of failure to the time of recovery must be less than the RTO specified for the application. Techniques for moving the recovery point (the right vertical bar) closer to the point of system failure, thereby minimizing system downtime, as shown in Figure 1b, is also a focus of this series.

RPO specifies the maximum amount of time between a backup of the database (full or incremental) and the point of failure, assuming that the database is active. For instance, if RPO is four hours, database backups (full or incremental) must be taken more frequently than every four hours. If RPO is five seconds, data replication is required; and the latency of the data-replication engine cannot exceed five seconds.

RTO specifies the maximum amount of time from the point of failure to system recovery. Having to repair a system in order to restore processing services can result in RTOs measured in hours or even days. Failing over to a backup system can reduce RTO to times ranging from hours down to minutes. And as we shall see, failing over to an operating backup system can lessen this time to seconds or subseconds.

Data-replication technology allows systems to be configured to meet any required RPO and RTO, as we will discuss later. In the extreme, zero RPOs and RTOs measured in subseconds can be achieved. But first, we explain how data replication works.

## Summary

A system may be down either because it is undergoing planned maintenance or because it has suffered an unplanned outage. In many systems, planned outages must be eliminated.

With respect to unplanned outages, the availability of a system can be characterized by several parameters, including availability, recovery time (RTO), and lost data (RPO). Systems can provide high availability (minutes of downtime per outage) or continuous availability (seconds of downtime per outage). Distributed systems can provide disaster recovery, in which operations can be restored following a disaster that destroys the production system, though this may take days or weeks. Extended distributed systems can provide disaster tolerance such that disasters have no effect on the end users.

The fundamental technology used to provide high- and continuous availability is data replication. In Part 2 of this series, we look at the various data-replication techniques available today.