

The digest of current topics on Continuous Processing Architectures. More than Business Continuity Planning.

BCP tells you how to *recover* from the effects of downtime.  
CPA tells you how to *avoid* the effects of downtime.

**In this issue:**

[Case Studies](#)

[How Does Google Do It?](#)

[Never Again](#)

[What? No Internet?](#)

[Availability Topics](#)

[Time Sync in Distributed Systems - Part 3](#)

[The Geek Corner](#)

[Hardware/Software Faults Revisited](#)

---

Complete articles may be found at  
<http://www.availabilitydigest.com/articles.htm>.

Earthquakes. Tsunamis. Power grid failures. Nuclear attacks. These are disasters that can suddenly disable entire business communities with no warning. But one disaster that we don't seem to worry about is a massive loss of the Internet. How many enterprises today depend upon the Internet for their day-to-day operations? How will they survive if they suddenly don't have Internet service?

It can't happen, many will say. It never has happened. Well, yes, it can; and yes, it has. Our Never Again story tells of a recent outage that suddenly disconnected a major part of the globe from the rest of us for days. Lightning never strikes twice? Wrong. Two days later, another Internet outage made a further massive impact on the same region.

Recovery from the loss of Internet services is too often not a part of our business continuity planning. Every enterprise should review its disaster recovery plans to ensure that contingency plans are in place to either provide backup Internet connectivity or to continue effectively without the Internet.

Global active/active data centers are one solution to this problem. Running in "split brain" mode during a massive disconnect is better than not running at all. This is one of the many topics we cover in our seminars that we tailor for companies such as yours. Contact us at [editor@availabilitydigest.com](mailto:editor@availabilitydigest.com) for further information.

Dr. Bill Highleyman, Managing Editor

---

## Case Studies

### How Does Google Do It?

Search for “availability” on Google. How does Google give you 674,000,000 web references in 150 milliseconds? Through an extremely efficient, large-scale index of petabytes (that’s millions of gigabytes) of web data.

A major problem Google faces is how to process this massive amount of worldwide data in a time that makes the indices useful. Google has solved this problem by building an equally massive parallel computing facility driven by its *MapReduce* function. MapReduce distributes applications crunching terabytes of data across hundreds or thousands of commodity PC-class machines to obtain results in minutes.

With MapReduce, Google programmers without any parallel and distributed systems experience can easily utilize the resources of these massive clusters. MapReduce takes care of partitioning the input data, scheduling the program’s execution across the machines, handling balky and failed machines, balancing the load, and managing intermachine communications.

Google fellows Jeffrey Dean and Sanjay Ghemawat describe the MapReduce function in their paper entitled MapReduce: Simplified Data Processing on Large Clusters. This Availability Digest article summarizes their description.

[--more--](#)

---

## Never Again

### What? No Internet?

On Wednesday, January 30, 2008, North Africa, the Middle East, and India experienced a massive Internet outage that was destined to last for several days or even weeks. How did this happen? How did companies cope? Could it happen in other areas such as Europe or the United States?

The bulk of data traffic from North Africa, from the Middle Eastern countries, and from India and Pakistan is routed through North Africa. There, it is carried by a set of three submarine cables that lie under the Mediterranean Sea. The cables link Alexandria, Egypt, with Palermo, Italy, where the traffic then moves on to Europe, the UK, and the Eastern United States.

On January 30, 2008, two of these three cables were severed. It is not yet known why, but the predominant theory is that the cables were severed by the anchor of a huge freighter. The result of this catastrophe was that 75% of channel capacity was lost from the Mideast to Europe and beyond.

[--more--](#)

---

## Availability Topics

### Time Synchronization in Distributed Systems – Part 3

In distributed systems, it is often imperative that all nodes in the system have the same view of time and that this time be synchronized with real civil time. In the Internet world, this is typically done with the Network Time Protocol, NTP. In Parts 1 and 2 of this series, we described the processes by which NTP accurately synchronizes every node in a distributed system with a reference civil-time source.

However, in many cases, we can relax the time requirements provided by NTP. Specifically, though it is imperative that the clocks of all nodes be synchronized with each other, it is often not necessary that the network be synchronized with civil time. If the network time is a few seconds or even more in error, this may not cause a problem so long as all nodes agree upon a common time.

Logical clocks, first proposed by Leslie Lamport in a 1978 paper published in the Communications of the ACM (Association for Computing Machinery), provide just this function in a much simpler way than NTP. Lamport was awarded the prestigious Dijkstra Prize by the ACM for his work on logical clocks.

[--more--](#)

---

## The Geek Corner

### Failure State Diagrams – Hardware/Software Faults Revisited

In an earlier article entitled [Calculating Availability – The Three Rs](#), published in December, 2006, we derived the failure probability of a system that must undergo repair, recovery, and restore operations. In that article, we assumed that the mean time to repair a node involved both node repair and node recovery. However, a node only requires repair if there has been a hardware failure. Many system failures are caused by node failures induced by software bugs, operator errors, or environmental faults. These sorts of systems require only a recovery.

This issue was addressed in our article entitled [Calculating Availability – Hardware/Software Faults](#), published in January, 2007. There, we arrived at a very simple and intuitive solution to the hardware/software problem. However, as this article shows, intuitive solutions can meet unexpected pitfalls.

Here, we take a formal approach to the hardware/software problem using failure state diagrams. We investigate the errors associated with the intuitive approach and conclude that it is valid for small values of system restore time (the time that it takes to restore system operations once a failed node has been returned to service). Derived is an accurate representation that is valid for large system restore times.

[--more--](#)

# Would You Like to Sign Up for the Free Digest by Fax?

Simply print out the following form, fill it in, and fax it to:

Availability Digest

+1 908 459 5543

Name: \_\_\_\_\_

Email Address: \_\_\_\_\_

Company: \_\_\_\_\_

Title: \_\_\_\_\_

Telephone No. \_\_\_\_\_

Address: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

The Availability Digest may be distributed freely. Please pass it on to an associate.

To be a reporter, visit [www.availabilitydigest.com/reporter](http://www.availabilitydigest.com/reporter).

Managing Editor - Dr. Bill Highleyman [editor@availabilitydigest.com](mailto:editor@availabilitydigest.com).

© 2008 Sombers Associates, Inc., and W. H. Highleyman