the Availability Digest
www.availabilitydigest.com

What Is Reliability?

June 2010

Andy Bailey, Stratus' Availability Architect, started a very active thread entitled "What does fault tolerant mean to you?" on our LinkedIn Continuous Availability Forum. Andy opened the thread with the observation:

> "I've long been aware that misuse of the term 'Fault Tolerance' (FT) is laying companies open to business risk and financial losses from system downtime."[1]

The range of responses support the general feeling that terms such as 'fault-tolerant,' 'high availability,' 'continuous availability,' 'mission critical,' and others serve the purpose of the writer, not the industry. Even characterizing system reliability in terms of 'nines' has its shortcomings since what is being included as downtime must be carefully defined.

In this article, we suggest a simple method to *quantify* system reliability, eliminating any ambiguity or suggestion of marketing intent in the above terms. Our result is a table that compares different highly-reliable technologies for IT systems. We encourage you to challenge and to comment about our suggestion on the Continuous Availability Forum by adding to Andy's thread.

Slushy Terminology

In the literature, the terms that are used to describe computer system reliability are what we might describe as "slushy." There are no agreed-upon definitions – the user of such terms is free to match them to his purpose, often with no attempt at a formal definition. There are certainly no legal definitions that would stand up in court.

Andy throws down the gauntlet with the following published statement:[2]

> "Fault tolerant computing is the ability to provide highly demanding enterprise application workloads with 99.999% (five nines) system uptime or better, zero failover time and no data loss…. 60% of IT users don't understand the difference between availability, high availability, and software-based fault tolerance, none of which meet the definition of fault-tolerance, and hardware-based fault-tolerance, which does."

Here are some of the observations posted to this thread:[3]

- [Continuously available] systems are available virtually all of the time – generally 99.999% of the time (about 30 seconds of downtime). Extensive use of independent

[1] See Andy's Availability Advisor blog at http://availabilityadvisor.com/.
[2] Stratus Technologies takes zero tolerance stance on misuse of term 'fault tolerant,' November 14, 2009 (http://www.stratus.com/news/2009/documents/20091112.pdf).
[3] Thanks to Randall Becker, George Ludgate, Tuomo Stauffer, Richard Buckle, and Moore Ewing for their comments posted to this thread.

© 2010 Sombers Associates, Inc., and W. H. Highleyman
www.availabilitydigest.com
For discussion, contact editor@availabilitydigest.com

components allows these systems to operate virtually without any unplanned outages. Planned outages do occur for upgrades, but the window for these outages is very small.

- [A fault-tolerant system] is a system that can survive any single fault. Of course, that bypasses the issue of what "survive" means. If you recover from the fault in five minutes, do you survive it?

- I distinguish between fault-tolerant systems and continuous availability systems, such as active/active systems. A true FT [fault-tolerant] system survives any single failure within the system. A CA [continuously available] system provides continuous uptime to its users, no matter what. The CA system may hiccup while it fails over, but if the hiccup is so short that no one notices, the system can be considered to be continuously available. The big difference is that a CA system keeps going, no matter what – even in the presence of a site failure. An FT system is not inherently a CA system because it does not protect against a site failure.

- A lot of my time is spent trying to understand the technical definitions companies are using in their literature. Do they make this easy – NO! … Where are we in having precise definitions of our availability terms for hardware, software and entire systems?

- It's a marketing term and it depends. Tandems (ok, HP NonStop) are great for recovering in transaction environments, really tolerate a fault and can be built [to] tolerate even more than two-way faults. But - fault tolerant can be thought many ways, what is a fault? We lost some deals because a security incident was thought to be a fault …

- Gartner Group, The Standish Group and even Jim Gray back in the old Tandem days included anything that can take a system down as a fault - from a hardware failure to an application bug.

- The problem may be the short-term horizon managers now work under. In many companies, the quarterly finances are far more important than the long-term effect of decisions. … If I have a failure twice a year, and 90% of those are recovered within my four-hour RTO [recovery time objective], an SLA [service level agreement] violation will occur on the average only once every five years. I'll probably be long-gone by then, promoted into higher management because I managed costs so well.

- A fault tolerant system is one that can survive a single point of failure. But should we cast the net wider, and include any outage – planned and unplanned? … In the end, if you think your system failed, then it did! If you believed you had an outage, you did! Perception is everything. Users looking at a blank screen for any period other than a screen refresh cycle, are witnessing an outage, surely!

- In general I think there is consensus that as long as a 'fault' doesn't materially affect a user then it could be interpreted as ft - to me this also includes no loss of any in-flight data.

- It further comes back to the question of what is Fault Tolerant. In my opinion, FT means that the corporation is insulated from any reasonable infrastructure fault (including applications/IT offerings in 'infrastructure', like phones, and power).

- An interesting pair of terms … is "fault tolerance" and "failure tolerance." … Fault tolerance [is defined] as "hardware that is built with redundant components to ensure that processing survives failure of an individual component." … Failure tolerance [is defined] as "an application that can continue even when failures such as node or site outages

© 2010 Sombers Associates, Inc., and W. H. Highleyman
www.availabilitydigest.com
For discussion, contact editor@availabilitydigest.com

occur." I interpret this as failure tolerance requires failover to a backup, but fault tolerance does not. … Clearly, failure tolerance requires an additional parameter and that is failover time. … All this leaves the question of the elimination of planned downtime up in the air.

- No technology … can prevent stupidity, can fix bad design, can always work, is suitable for everything … is the best … solution for everything, etc. But, again … anytime a user/customer sees the system "down" - it is down, it doesn't matter where the problem is.

- Most terms lack a legal, formal, tight definition and even when they do there is little the owner can do if the public decide to use it differently. A working definition might be "a f-t system is one with no SPOF which causes the system to cease to function correctly". Any such system will be composed of multiple subsystems all of which may use very different techniques to remove SPOFs. The problem arises when technical terms become marketing slogans.

- The most "legal" definition typically comes from the service level agreements. … In a contract, one might put "Mission Critical is defined as being available x% of the time over t years" to provide that definition.

- For terms like "Mission Critical" or fault-tolerant", I don't think there needs to be a legal, or even a formal, definition. Neither has any place in the requirements specification of a development. Both relate to the real requirement "Availability". One [mission critical] is a vague description of an availability requirement, the other [fault-tolerant] of an approach by which it will be achieved.

The final comment seems to sum up the sentiment. Reliability terms in use today are either vague definitions of reliability requirements or are vague descriptions of a reliability approach.

How Do We Quantify Reliability?

We submit that 'reliability' is a measure of the user experience. Therefore, at the risk of attempting to define terms that are not inherently definable, we specify for purposes of this paper that 'reliability' is a measure of a user's experience with respect to the provision of IT services to him. It is this measure that we want to quantify.

Recovery is the Key

Shimon Peres made the observation that

> "*if a problem has no solution, it may not be a problem, but a fact, not to be solved, but to be coped with …*"

Failure is a fact. Recovery is how we cope with it.

We are stuck with component failures. We can't control failure rates (except to buy better components). What we can control is how we get back into business when a component fails – how do we recover from the fault? Reducing recovery times by a factor of ten has the same impact on availability improvement as increasing failure intervals by a factor of ten.

There are two aspects to the recovery of an IT system – recovery of services and recovery of lost data. For these two aspects, there are well-defined terms in the industry to quantify them – RTO (Recovery Time Objective) and RPO (Recovery Point Objective). Though technically they are objectives to be set forth in an SLA, we can also view them as descriptors of system reliability.

© 2010 Sombers Associates, Inc., and W. H. Highleyman
www.availabilitydigest.com
For discussion, contact editor@availabilitydigest.com

RTO is the time that services are expected to be down. RPO is the amount of data that may be lost and must be recovered.

So why not characterize the reliability of systems by their expected RTOs and RPOs? True, one could still play with failure scenarios and choose those that give a better marketing image. But we can scope RTOs and RPOs to a fairly narrow range and thus obtain reasonable comparisons between systems.

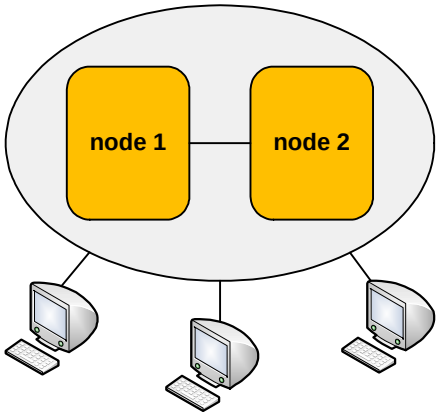
But Reliability is More Than RPO and RTO

If we accept that reliability is a measure of the user experience, simply measuring a system's mean time to recover and average amount of data lost following a failure does not tell the whole story.

Site Disasters

The literature is rife with systems being taken down for hours or days due to a data-center disaster. Does this impact a user's experience? You better believe that it does. Therefore, the ability of a system to survive a site disaster is another key parameter.

Reliable systems are always redundant. The question is, how far apart can the redundant systems be separated so that a common disastrous event does not take down both systems? We therefore include a distance parameter in our specification as a measure of disaster tolerance.



How fast can I recover?
How much data will I lose?
Can I survive a disaster?
Can I eliminate planned downtime?

Planned Downtime

Systems must be taken offline occasionally to upgrade hardware, operating systems, database management systems, applications, and so forth. Does this impact the user experience?

It depends. If the system is not a 24x7 system, and if there is a sufficiently long window during which the system can be taken down to be upgraded, then planned downtime is nonintrusive (assuming that the system comes back online properly following the upgrade). However, if the system must be in continuous operation, then planned downtime must be eliminated.

The typical way to eliminate planned downtime is to roll the upgrade through the system one node at a time.[4] Some systems allow this, and others do not.

We therefore add another parameter that indicates whether upgrades can or cannot be rolled through the system.

Failure Rates

The availability of a system is a function not only of its recovery time but also of its failure rate. Given the same recovery time, a system that fails twice per year will have half the reliability of a system that fails once per year.

[4] Achieving Century Uptimes: Parts 15, 16 – Zero Downtime Migrations for Active/Active Systems, *The Connection*; March/April, May/June 2009.

© 2010 Sombers Associates, Inc., and W. H. Highleyman
www.availabilitydigest.com
For discussion, contact editor@availabilitydigest.com

How do we account for failure rate in our comparisons? We don't. Failure rate is a function of the components we choose to implement our system. It is not a fundamental characteristic of the architecture of the system or of its underlying technology.

Application Support

Some architectures support certain applications better than others. For instance, some architectures work best for database applications, whereas others are applicable to a broader set of applications. In addition, each architecture usually supports only certain vendor hardware, certain operating systems, and certain databases.

We leave this to the marketing people.

Nines

Note that we have left out 'nines' as an availability measure. If we are talking about highly-reliable systems, a measure of availability is, in our opinion, far less meaningful to the user experience than is recovery. Besides, a measure of 'nines' is directly dependent upon the reliability of the components of a system, a factor that we are ignoring on the basis that failure rates are less important if recovery is fast enough. If 'nines' are used, one must clearly state which downtime scenarios are included in the measure and which are not.

This is not to say that 'nines' are not an important measure of a system's reliability. We are only saying that we can judge the relative reliability of a system by measures other than 'nines.'

All Those Other Faults

Are we only comparing downtimes caused by hardware and software failures? What about all of those other causes of downtime – operator errors, application bugs, network faults, environmental faults? After all, with the reliability of system hardware and software today, they are often the predominant causes of system failure.

Some architectures inherently allow an independent recovery attempt from any of them. Others do not. This is a factor that must be considered, but is outside the scope of our qualification.

Cost

To soothe the savage marketing community, we leave cost comparisons to them. Beware of sales by under-configuration.

System Reliability Comparison – a Straw Man

The following table is a first pass at quantifying system reliability using the above concepts. By-and-large, the parameters (if accurate) cannot be easily manipulated for marketing purposes. It is fairly easy to add new systems to the list without extensive benchmarking – all that is needed is an understanding of the architecture.

There is no "best" system. It all depends upon the needs of the application. Hopefully, this sort of comparison can be a valuable tool in matching a highly-reliable technology to the needs of an application without being mislead and confused by loosely-defined marketing terms.

© 2010 Sombers Associates, Inc., and W. H. Highleyman
www.availabilitydigest.com
For discussion, contact editor@availabilitydigest.com

System	RTO	RPO	Disaster Tolerance	Rolling Upgrades
Stratus ftServer[5]	0	0	0	yes*
Stratus Avance[6]	seconds	0	0.5 km	yes
Marathon EverRun	seconds	0	meters	yes
HP NonStop DMR	msec.	0	0	no
HP NonStop TMR	0	0	0	no
HP NonStop Blades[7]	msec.	0	0	no
HP NonStop ZLT	minutes	0	tens of km	yes
OpenVMS cluster[8]	msec.	0	600 miles**	yes
Clusters[9]	minutes	0	0	maybe***
Active/Active Async[10]	seconds	seconds	unlimited	yes
Active/Active Sync[11]	seconds	0	tens of km	yes
Active/Active Coordinated Commits	seconds	0	unlimited	yes
IBM Parallel Sysplex – Metro Mirror[12]	minutes	0	100 km	yes
IBM Parallel Sysplex – Global Mirror[12]	minutes	minutes	unlimited	yes
Unidirectional Replication	minutes	minutes	unlimited	yes
Virtual Tape	hours	hours	unlimited	yes
Magnetic Tape	days	days	unlimited	yes

* With Active Upgrade feature.
** Significant performance degradation since synchronous replication is used.
***If operating system is on private disks, not public disks.

A Comparison of System Reliability

[5] Fault-Tolerant Windows and Linux from Stratus, *Availability Digest*; September 2007 (http://www.availabilitydigest.com/public_articles/0209/stratus.pdf).
[6] Stratus Avance Brings Availability to the Edge, *Availability Digest*; February 2009 (http://www.availabilitydigest.com/public_articles/0402/avance.pdf).
[7] HP's NonStop Blades, *Availability Digest*; August 2008 (http://www.availabilitydigest.com/public_articles/0308/ns_blades.pdf).
[8] OpenVMS Active/Active Split-Site Clusters, *Availability Digest*; June 2008 (http://www.availabilitydigest.com/public_articles/0306/openvms.pdf).
[9] Active/Active Versus Clusters, *Availability Digest*; May 2007 (http://www.availabilitydigest.com/private/0205/clusters.pdf).
[10] Asynchronous Replication Engines, *Availability Digest*; November 2006 (http://www.availabilitydigest.com/private/0102/asynchronous_replication.pdf).
[11] Synchronous Replication, *Availability Digest*; December 2006 (http://www.availabilitydigest.com/private/0103/synchronous_replication.pdf).
[12] Parallel Sysplex – Fault Tolerance from IBM, *Availability Digest*; April 2008 (http://www.availabilitydigest.com/public_articles/0304/ibm_sysplex.pdf).

© 2010 Sombers Associates, Inc., and W. H. Highleyman
www.availabilitydigest.com
For discussion, contact editor@availabilitydigest.com

Summary

If one accepts the premise that reliability is in the perception of users, and if that user perception is recovery time, then system reliabilities can be compared quantitatively.

The first step in any such evaluation is to define the reliability requirements of the applications (different applications will certainly have different reliability requirements). Various products and technologies can then be evaluated to narrow the scope of the search. Once a subset of applicable systems is determined, focus must then be placed on choosing those that support the systems and applications of interest. Last but not least, the deciding factor will be cost.

It must be noted that the cost of a system is not just a matter of acquisition cost or of operating cost. It is also a matter of downtime cost. With downtime costs varying from thousands of dollars per hour to hundreds of thousands of dollars per hour and more, the difference between recovery times measured in milliseconds, seconds, or minutes may well justify a more expensive system.

Let us have your feedback to this attempt at defining reliability on the Continuous Availability Forum.

© 2010 Sombers Associates, Inc., and W. H. Highleyman
www.availabilitydigest.com
For discussion, contact editor@availabilitydigest.com