the Availability Digest

www.availabilitydigest.com

Enterprise Availability Architectures for Business-Critical Services

Arvin Levine, PhD
June 2013

Abstract

Large enterprises continue to search for an approach to supporting business critical processing with the right balance of risk and cost. Industry experience shows (or at least an industry 'urban legend' has it) that companies that lose critical data processing capabilities for an extended time may be unable to ever recover from the impact of an outage. On the other extreme, it is easy to overspend on availability, whether as a 'belt and suspenders' approach to provide extra assurance of recoverability or merely to simplify the operational aspects of availability. Even overspending does not guarantee appropriate enterprise availability without a well thought out and architected implementation of applications, supporting technology and processes. This paper provides an approach to risk-based availability architecture and analysis.

Introduction

The past decades have seen a heightened awareness of the impact of disaster-scale data-processing outages on large enterprises[i]. Many enterprises have become largely 'information businesses' or at least critically dependent on information processing to enable key business functions. Customers, stakeholders and government agencies increasingly require these enterprises to secure the basic information processing resources needed to stay in business and fulfill obligations. For example, regulated businesses (e.g. banks) must satisfy auditors that their recovery processes and plans will enable them to continue or recover normal operations within a relatively short time following a recognized disaster.

> The CNN Moment is that agonizing disaster where the dot-com infrastructure fails and its architectural shortcomings end up as a lead story on CNN's financial news. As everyone battles to restore the operational integrity of the site, the hapless company combats angry customers, disappointed analysts, falling stock prices, reduced earnings, lawsuits, and a plethora of other ills that can damage the value of the dot-com brand and, in some cases, imperil the company's very survival. At a minimum it is always the kind of bad publicity that every dot-com wants to avoid.
>
> From J. Williams, Avoiding the CNN Moment, IT Professional, Volume 3 Issue 2, March 2001

Disasters impacting enterprise IT may come in many flavors. They may be both natural and man-made and may also include self-inflicted situations. Some examples will illustrate the range of events:

- Tsunami, hurricane, tornado – destroys buildings and/or cuts electric power and communications in a geographic region
- Terrorist attack, computer virus – destroys a facility, sabotages equipment, deletes data
- Human error -- a back-hoe driver cuts a key cable, an operator inadvertently (intentionally?) turns off a key power switch, a programmer does a 'delete all' command in the wrong context

In an information business, disruptions to infrastructure or applications can have measurable business costs, ranging from small change for a dropped transaction to millions of dollars when business is stopped for extended periods of time. Even a small disruption can incur significant cost to determine whether (and how to correct) any data corruption that may have resulted. Other dimensions of disruption (failure to data integrity or confidentiality) may also be costly. For example, a disruption of data integrity due to added zeros (initiating a trade of millions of shares rather than thousands) recently sent the global stock market spinning. Similarly, disruption of data confidentiality could cause significant loss of customer confidence, loss of competitive advantage, or adverse market price fluctuations. Regulatory fines may be imposed as well when instances are deemed to result from the institution's negligence. A common version of the disaster professional's urban legend contends that, for companies that had a major loss of business data, 43% never reopen and 29% close within two years. While this claim is largely unverified[ii], it captures the industry perspective nicely.

Responding to the recognition that availability and disaster recovery are not simply luxuries for information technology, institutions now implement broad measures to protect all their IT assets. What had been a niche business for areas that had clear business justification (e.g., ATM/cash machine networks) has become a standard data center practice or infrastructure-based service offering. Together with the increased technological support for availability, applications increasingly define themselves as requiring higher levels of business critical support, generally expressed as high availability and disaster recovery. This 'white glove' treatment is now the norm in many information industries, particularly the finance industry.

Often, even though applications define themselves as requiring high availability, they do not adapt their programming models to support this. It is simpler for an application to leverage support services in the infrastructure, such as duplicated disk storage, replicated SAN storage and heightened data center-wide backup and recovery. Processes and failover tests are used as after-the-fact assurance that the measures adopted are satisfactory or can be remediated. Infrastructure, in turn, is often optimized to seamlessly provide these capabilities with no direct connection to the actual business or application requirements which invoked them in the first place. Ultimately, the proverbial cart is often placed in front of the horse, with the data center needs taking on a life of their own, regardless of the application or business requirements. A trend like this can result in costly, excessive and ineffective investments in availability -- almost as bad a situation as the opposite extreme of no availability support.

At this point, architects might well interject themselves into the topic. As depicted in Figure 1, the goal of an architected approach is to provide the most cost effective strategy, while offering the needed level of protection.

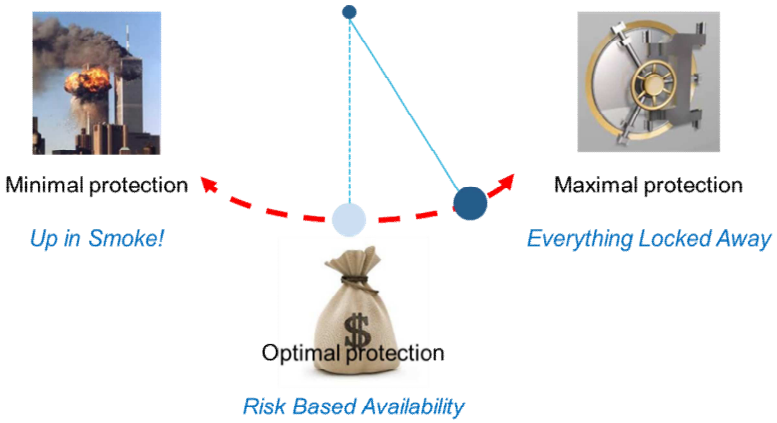

Figure 1: Achieving optimal availability

The architecture approach is to analyze risk based availability – the alignment of business risk with technology capabilities:

- categorize the business requirements and classify applications,
- analyze the technology capabilities, methodologies and costs,
- estimate investment and cost requirements,
- plan the evolution from current to desired.

Following a brief definition of availability, the rest of this paper will discuss these architecture activities in more detail.

Defining Availability

Availability can take a range of characteristics and can be viewed from multiple technology and business viewpoints. For example, is an application available if it can only process half its projected volume (as illustrated in Figure 2, below)? What if that happens after a declared event (disaster)?

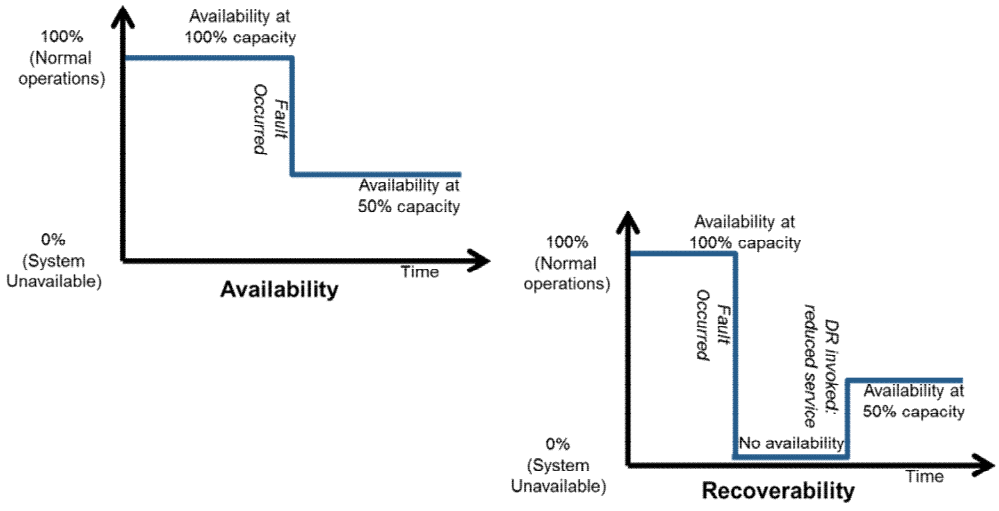

Figure 2: Availability and Recoverability

Similar questions can be asked about data availability. Is data available if it is out of date, corrupted or of uncertain quality? Is data actually available if you can only retrieve it together with extraneous material, which may confuse or delay its recovery? What about rarely accessed historical data?

While there is no single right answer for these questions, we provide a basic definition for availability which can then be adjusted to suit specific business conditions.

Definition: An application or system is *available* when it can correctly process its normal expected volume of business. It is *recovered* from a declared fault when it can reach that same processing criterion. *Data is available* only if it has no corruption or inconsistency; a subset of data may be considered available, even if the full dataset is not.

While not the topic here, we note that an application or system that is not able to handle a spike in volume might also be considered to have availability issues even if nothing is actually malfunctioning.

Categorize the business requirements

Much of the IT industry has realized that availability and disaster recovery are requirements for applications that are significant to the business. What was once the province of niche applications and vendors (Tandem/NonStop, Stratus/S88, Dec-FT, with their warring "5-nines" of availability claims), now applies across the board to all business applications and areas.

A typical way to communicate the business requirements for availability is to specify a level of criticality: High/Medium/Low. While this terminology may be intelligible to the business, it is far from obvious what requirement this places on an application. Also, there is more than a single dimension to availability. Commonly used technical descriptors, including "Mean Time Between Failures" (MTBF), "Mean Time to Repair" (MTTR), "Recovery Time Objective" (RTO), and, "Recovery Point Objective" (RPO), are deeply technical in nature and not obvious to the business[iii].

The classification terminology for business requirements should be natural, but fairly precise to satisfy both sides of the communication. A framework for this terminology would be to define an "Availability Class" definition (AC-n) with both business criteria and technical specifications. Each AC would cover a defined range of cases that would not overlap with the other classes. The granularity of such an AC system depends on the nature of the business requirements:

- A three tier (AC-1, AC-2, AC-3) system would intuitively correspond to high/medium/low availability.
- A four tier system (AC-1, …, AC-4) could add a 'super' high availability rating.
- The middle tier could be subdivided for different (data) recovery times with the same basic outage class of service.
- An extra tier could be defined to specify the characteristics of a unique business or technology environment, e.g., high-frequency trading or remote office applications, or, using cloud or co-located technology, etc.

For these tier definitions, disaster recovery and high availability service levels are treated as one subject, but they could be treated as distinct business and technology problems. In general, disaster recovery can be included in the availability service tier, but it may also entail wider support issues, such as remote or secondary access, and, transportation and housing for alternative venues. The wider scope might raise the need to classify applications for these requirements separately from the availability tiers.

As one such example, we present a table (Figure 3) for a three tier classification, specifying the AC, key technical criteria and business criteria.

Availability Classifications				
Availability Class	Annual Availability	Maximum outage per incident (RTO)	Maximum Data Loss (RPO)	Business Criteria
AC-1	99.9%	4 hours	Minimal to none (possible minutes of data lost)	Regulatory requirement, legal issue, documented extreme business value
AC-2	99.4%	24 hours	Most recent 24 hours of data may be lost	Supports critical business areas
AC-3	Best effort	n/a (best effort)	Best effort, no guarantee	Risks must be accepted by the business

Figure 3: Availability classifications

In the table, AC-1 specifies "3-nines" of availability. This means that a business owner can expect to provide service with almost no outages (total annual outage time measured in minutes to hours). RTO of 4 hours means that any outage is resolved in less than a business day, while the RPO means that only the most recent activity could be lost or corrupted as a result of an outage. AC-1 is an expensive service. Generally, this is the highest class of service that an enterprise would provide and should be restricted to business areas that can demonstrate the value or requirement for this sort of service. Typically, transactions that capture monetary obligations (sales, orders, trades, payments, etc.) and that occur in an online real-time fashion are candidates for this service. Higher service levels (e.g., more "nines", quicker RTO, continuous availability with no data loss) could be defined either as a subset of this class or as a separate availability tier, if there was a specific business requirement in this area.

AC-2 would usually be considered 'standard' service. A good expectation of availability (hours to days of outage per year) is specified, with outages resolved by the next day and data recovered from the end of the previous business day. This service class is generally applicable to any application whose business value is not seriously disrupted if the process is stalled for a day. Most reporting functions and analysis activities can tolerate this level of availability.

AC-3 represents a 'best efforts' service level intended for utility or internal applications which can tolerate being at risk of failure and moderate data loss. Even these applications will have a fairly high level of availability service, simply because standard modern technology and processes have a high level of quality built into them. Applications in this tier would receive the lowest prioritization during any recovery activity. Note that a business could architect AC-3 as their standard infrastructure support and require applications to engineer their own higher availability using software techniques. An example of this was the approach provided to Amazon Web Services customers, where lowest cost for basic services was a key requirement.

There could well be special case exceptions to the tiering system. For example data-warehouse applications may be regarded as critical to the business, but not worth the extreme cost required to put them in AC-1. Furthermore, data loss is not an issue for these systems, in general, since they are consuming information generated from the enterprise which may most efficiently be recreated from the source systems rather than restored from a data-warehouse replica or back-up. On the other extreme, market data systems are viewed as critical in financial enterprises, but may not have the RTO (data loss) requirement since the data becomes historical and of low value in a very short time. While counter-intuitive, the AC-2 or even AC-3 rating might be more appropriate for these systems!

This discussion has illustrated how there is value in adopting a tiered availability classification for business applications. The assignment of applications to the tiers requires business participation, working together with the applications development team and architects, just as for any other set of business requirements. Having a mutually understandable classification terminology facilitates that effort.

Analyze the technology capabilities, methodologies and costs

There are many technological solutions that can be applied to availability requirements. At an intuitive level, all processes and data could be duplicated (NASA at one point implemented triple redundant systems with a process for voting on the 'truth' for space missions). This 'one size fits all' approach is both expensive and unsatisfactory. It is expensive since the basic cost is doubled for system components. It is unsatisfactory since having more complexity adds to the probability of error and outages of some key components without guaranteeing basic integrity of the results. Furthermore, if an error occurs, the duplicated data would also contain the same error leaving the 'back-up' as useless as the main system!

A certain level of system component redundancy is vital to maintaining availability, and this can be adjusted to provide the tiered availability as discussed above. In addition, data back-up strategies, application development patterns, software layering, and monitoring systems can be deployed as well.

The table in Figure 4 provides some implementation examples corresponding to the sample availability classes outlined in the previous section. Each strategy carries an investment, not only for capital hardware and software, but also for ongoing support and operations. In general, the highest availability class (here AC-1) has the largest number of implementation strategies and can be supported at different investment levels and for different technical environments.

Availability Implementations			
Availability Class	**Maximum outage per incident (RTO)**	**Maximum Data Loss (RPO)**	**Sample Implementations**
AC-1	4 hours	Minimal to none (possible minutes of data lost)	• Distributed cluster • Active-Active multi-site availability • Dedicated specialty platforms • Replicated Virtualization • Hot/cold standby servers • Subsystem or software mirroring • Database replication • Network level mirroring
AC-2	24 hours	Most recent 24 hours of data may be lost	• Allocated hardware • Backup data restore • Restore application and data from near-line backup
AC-3	n/a (best effort)	Best effort, no guarantee	• Restore application and data from near-line backup

Figure 4: Availability implementation examples

While the table looks like a 'Chinese restaurant menu', where the architect can select solutions at random, a better approach is to align the applications and infrastructure platforms. Aligning the layers of the architecture can help avoid design and operational errors that result from unfulfilled assumptions. Having a consistent set of platforms reduces the application development complexity and learning curve, as well.

It is not necessarily the case that the higher investment for an AC-1 is 'wasted money' even without the availability protection. Consider that an active-active environment can be used to support faster response times or higher throughput volumes when not required for availability. Similarly, database replication can enable secondary use of the data at a back-up site, for example, reporting or query activity. Agreement would be needed that these secondary uses will not be provided during recoveries. A higher availability environment will cost more due to more expensive components and higher design costs. There is no 'free lunch', even though the differential cost for high availability may not be as significant as in the past.

It is also possible, if implementing a 'green field' environment, to set the entire application development and system support approach in a high availability framework. There exist special purpose systems that embody both the infrastructure elements and the programming model (e.g., Tandem/NonStop) required, but a similar effect can be achieved with a custom designed technical architecture, especially using modern virtualization and resilient (web) software components and monitors.

Estimate relative cost

The table in Figure 5 presents high-level application cost estimates corresponding to each of the availability classes that were defined above. The calculations used are fairly simple and transparent, but experience has shown that they correspond well with more detailed and rigorous analyses. The cost was calculated as the sum of the major implementation components that provide the availability characteristics in that class. The model of availability used for this calculation assumes maximum back-up via duplication of resources, including disaster recovery (DR).

Availability Cost Estimate			
Availability Class	**Abstract cost estimate**	**Assumed implementation components**	**Discussion**
AC-1	13.5	Servers: 4x Data: 8x Data Center: 1.5x	• Duplicated servers & data for availability • Duplicated back-up of data • Duplicate environment for DR • Additional operations, networking • Most resilient Data Center (Tier IV)
AC-2	7.5	Servers: 2x Data: 4x Data Center: 1.5x	• Duplicated resources for local availability • DR resources shared with 'stoppable' activities, e.g. Application development • Additional operations, networking • Resilient Data Center (Tier II/III)
AC-3	2.5	Servers: 1x Data: 1x Data Center: .5x	• Best efforts with no explicit duplication • No explicit DR recovery capacity • Standard operations, networking • Standard Data Center (Tier II)

Figure 5: Availability costing model

Regardless of the precision of the model, certain reasonable implications can be drawn from the estimates in the table. There is certainly a large multiplier effect between the availability support options for a given application. In the table, AC-1 is roughly 5 times more expensive than AC-3, and, nearly twice as expensive as AC-2. This is intuitively reasonable, but gets additional force from being quantified. Business and application organizations are likely to consider a cost differential of that scale when deciding on the availability requirement for their projects.

Another implication from this table might be that complete duplication of resources is a very expensive approach to providing high availability and disaster recovery. While such strategies are generally the simplest to implement, we discussed above that they may not securely provide the availability desired, despite the expense. A more detailed risk-benefit analysis could compare other alternatives, including N+1 based availability (providing spare capacity that could be applied as needed), and, active-active environments (temporarily reducing capacity following a failure). Alternative availability architectures bring additional complexities and costs of their own to an organization, but they could be less expensive when viewed from the enterprise perspective.

Adding a quantitative measure to the availability discussion both increases the impact of a risk based availability analysis and permits the business to understand the risk and reward equation better.

Plan the evolution

The architecture approach advocated here for aligning business risk with technology capabilities uses an analysis that enables implementation. Ultimately, the goal of the entire approach is to reduce cost while continuing to provide appropriate support for the business. After selecting appropriate availability categories with associated implementation technologies and investments, the execution steps are:

- business classification,
- technical architecture and design, and,
- operational process and technology implementation.

The implementation of **business classification** requires that business owners assign the availability classes to their applications so that technical requirements can be appropriately formulated. The assignment of availability classes can potentially uncover unrelated issues, such as project/technology ownership uncertainty (multiple business owners, no business owners) and missing application component inventory tracking or control. This can be viewed as either an opportunity or a distraction! In any case, the ability to provide a high-level cost model for different possible availability scenarios can enable the business to appropriately judge the risk it is prepared to accept. The technology teams would participate in this activity to ensure that reasonable expectations are being agreed upon throughout the classification exercise, especially in conjunction with the other application requirements and timeframes.

Technical architecture and design involves the application-level, infrastructure, and operations organizations. Planning and architecture is key, both to avoid costly multiplication of methods and technologies, but also, to enable application design to be aware of, and to take advantage of, these capabilities. Technical capabilities for availability must be incorporated into an organization's overall technology roadmap. For example, availability is only one of many drivers of a database platform strategy; server virtualization and cloud can also include provisions for availability support and disaster recovery. Applications and the development process should be aware of availability requirements. For example, a web portal could include automated re-routing of failed messages; a store-and-forward message bus could incorporate an out-of-band restart of target servers in addition to re-delivery of messages; a transaction monitor could start additional or remote servers and redirect traffic.

Operational processes should take account of the differentiated availability classifications. Data-center wide back-ups may prove to be less useful tools as compared to transactional replication of audit logs for selected applications. Archival and hierarchical storage concepts could also be employed. In addition, prioritization of application support may be required to reduce costs while providing agreed upon service levels. Access for personnel to facilities, as well as command and control capabilities for both infrastructure elements and applications, need to be planned.

Technology implementation can face a 'chicken and egg' situation, where the infrastructure components are out of step with the applications or operations in regards to availability implementation. Specific strategic investments and programs may be needed to overcome this gap. For example, an

infrastructure capability might be implemented (prior to application users requiring it) in order to be the standard support for new applications with the appropriate availability requirement.

Conclusion

Architected availability and disaster recovery are undoubtedly significant aspects of the compute strategy in any large enterprise. A risk-based analysis of the requirements and potential solutions can result in differentiated implementations, resulting in significant cost savings over simple replication-centered implementations, while providing better levels of support for the organization's actual needs. Understanding availability implications enables business and technical communities to define and fulfill appropriate application and infrastructure requirements. Architectural methods, including categorization of availability requirements, analysis of relevant technology and attendant costs, and planning processes and implementations, are the main steps in achieving the appropriately balanced solution, as depicted in Figure 6.

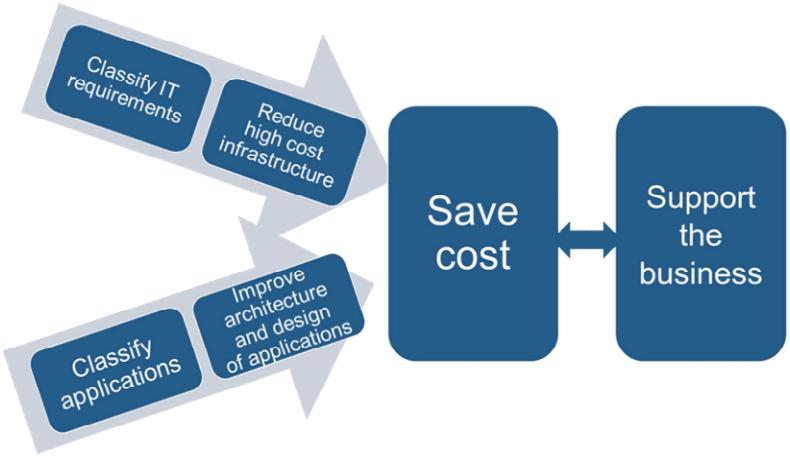

Figure 6: The goal of a risk-based availability analysis

A reasonably balanced, well architected availability environment can be achieved and can be a valuable part of the overall enterprise IT portfolio.

Author's Biography

Arvin Levine has been involved in architecting high availability and disaster recovery solutions for large enterprises, originally as part of the Tandem (aka HP NonStop) pre-sales organization, and, later as a senior infrastructure architect at Credit Suisse. This article draws on insights from many engagements, and, especially, from working with former colleagues and SME's, too numerous to name, who have "drunk the kool-aid" of architecture and availability and are committed to bring more of both into the world. Arvin can be reached at arvinlevine@gmail.com.

---

[i] A related paper was presented at **CEWIT2011** as "*Defining a Risk-based Approach to the Design and Technology Usage of Systems to Attain IT Availability and Recoverability Requirements*", by Michael Azzopardi, Arvin Levine and John Lamb, November 3, 2011.

[ii] Business continuity statistics: where myth meets fact. , Mel Gosling and Andrew Hiles, Continuity Central. 24 April 2009. Retrieved 3 August 2012. Also see Avoiding the CNN Moment, J. Williams, IT Professional, Volume 3 Issue 2, March 2001, Page 72, 68-70 EEE Educational Activities Department Piscataway, NJ, USA doi>10.1109/6294.918228, and, **Breaking the Availability Barrier** (3 volumes), Bruce Holenstein, Bill Highleyman and Paul J. Holenstein. Author House (2003, 2007, 2007). ISBN 978-1410792327, 978-1434316042, 978-1434316073

[iii] Indeed, these terms often confuse practitioners, as well. For example, MTBF and uptime were conflated in an earlier draft. They are related: Uptime (or annual availability) is the proportion of time that the system is available, usually expressed in 9s. A system with an MTBF of six months and a recovery time (RTO) of four hours has an uptime of three 9s - it is down eight hours per year.