TECHNICAL MEMORANDUM

High-Availability Issues and Solutions: An Overview

David M. Ihnat
dihnat@dminet.com

January, 2004

1. INTRODUCTION

A number of schemes exist to implement hardware and software high-availability solutions. Without endorsing any particular product or vendor, this paper describes what is meant when using these terms, and the current range of options available to provide such solutions.

2. OVERVIEW

The generic term *High Availability* has been used in the computer industry and academia to describe systems evincing a wide range of fail-safe or fail-soft capabilities. These may range from fully redundant, "non-stop" systems[1], down to systems that simply use RAID technology to provide protection of data and programs from damage or loss inherent in single-disk failure. Thus, it may be seen that any time this phrase is used, care must be taken to determine exactly what is meant in context.

In this document, we will consider High Availability to indicate incorporation some or all of the generally accepted approaches available commercially to provide fault tolerance to either an entire computing system, or elements within that system.

The degree to which any system is considered High Availability is driven by business requirements. The greater the degree of reliability required, the greater the complexity and cost. Often, a fully fault tolerant system is overkill, in both terms of cost and complexity, and the features provided.

Thus, a reasonable determination of the business requirements driving the request for a High Availability solution is critical to determining the best combination of features and cost effectiveness.

3. GLOSSARY

Please note that these definitions are derived from many sources, and the definitions provided here are the author's own synthesis.

Checkpoint When used in terms of a journaled filesystem, a recorded, known and consistent state to which the system can be recovered. Commonly

[1] Such as those marketed by Tandem (now owned by Compaq) and Stratus Computers.

Copyright ® 2004 David M. Ihnat

	used as the starting point for application of journal transactions; see *Journaling Filesystem*.
High Availability	A term describing a computing environment incorporating hardware, software, and operational practices designed to provide varying degrees of resistance to failure in any combination of these components.
Fail-Safe	An term applied to extreme High Availability solutions intended to provide continuing unattended operation under all foreseeable failure modes. See also *Non-Stop*.
Fail-Soft	A term applied to High Availability solutions intended to be fault-tolerant but not necessarily non-stop. Fail-soft systems may require operator intervention or repair to return to full operational status.
Fault Tolerant	A term used to indicate some degree of redundancy, self-repair, or other system provision to eliminate or mitigate the effects of hardware or software failure.
Filesystem Appliance	Another common term for a *NAS appliance*.
Journaling Filesystem	A type of disk filesystem that logs changes applied to data and directory structures to permit recovery from a known, saved point. See *Checkpoint*.
Mirroring	A type of RAID recovery in which an entire disk and/or partition of a disk is guaranteed to be copied *in toto* to two different physical devices, allowing recovery if one should fail.
MTBF	**M**ean **T**ime **B**etween **F**ailure. A measure of the anticipated reliability of a system or component, usually expressed in terms of hours of continuous operation.
NAS	**N**etwork **A**ttached **S**torage
NAS appliance	Common term for a network-attached filesystem device.
Non-Stop	Another common term for *fail-safe* systems. Derived from (and trademark of) a particular vendor that was active in early development of fail-safe systems.
RAID	**R**edundant **A**rray of **I**nexpensive **D**isks. A generic term for a family of data protection and recovery models distinguished by different methods of data duplication and encoding, and referred to by level, e.g., *RAID-5* vs. *RAID-2*.
Single Point of Failure	A term used to indicate a component or element of a system or infrastructure which will cause system failure despite any other redundancies or fault-tolerant elements.

4. SYSTEM ELEMENTS

Computer systems, for the purpose of defining fault-tolerance, may be considered to consist of four major subsystems[1]:

- *Server*. Those elements of the system providing software execution and control. Typically the CPU(s), memory, internal controller cards, internal power supply, etc.
- *Data Storage*. Those elements of the system providing data storage and retrieval services. Typically disks, disk communications, disk controllers, etc.
- *Communications*. Networking and/or other communications devices and infrastructure.
- *Power*. Production and distribution of power to all systems elements.

Following sections will discuss these systems in terms of issues and solutions related to providing varying degrees of fault tolerance.

5. SERVERS

Server reliability can be accomplished by three methods:

- Making a single server fully or partially fault tolerant.
- Guaranteeing a replacement server is available if the primary fails
- Making several servers share capabilities.

Of course, hybrid combinations of these approaches are seen in the field as well. All have, as their goal, reducing the cost and complexity of the system while maintaining an acceptable level of fault tolerance. That level must be determined on a per-situation basis.

5.1 NON-STOP/FAIL-SAFE SYSTEMS

Non-stop or *fail-safe* systems evoke memories of the hubris surrounding such disasters as the Titanic. However, unlike that allegedly unsinkable failure, it is effectively possible to build a computer system that will not fail, short of physical destruction. This accomplished by duplicating[2] every component of the computer considered critical to operation, from the UPS through the computer's bus. Essentially, every *single point of failure* has been eliminated from a non-stop system.

Such solutions are typically not built by end-users from components. The degree of integration, of both software and hardware, dictates proprietary systems provided by vendors specializing in such solutions. Examples of such systems are *Stratus' Continuum* and *ftServer* lines.

---

[1] Of necessity, there are simplifications in this model. For instance, if Data Storage is totally contained within the server, are the disk controller cards to be considered part of the server, or the data storage subsystem? Generally speaking, however, these definitions work well enough for practical planning purposes.

[2] In some implementations, even greater levels of redundancy are used, resulting in multiple power supplies, CPUs, controller cards, etc. Complexity of both hardware and controlling software, of course, spirals, as does cost.

These systems, however, are capable of achieving such a degree of reliability only with concomitant extreme expense and intrusive complexity. If the application truly requires such non-stop operation—business requirements are such that this cost is acceptable—then is can be borne.

In most cases, however, such a solution is neither affordable nor necessary. It may be acceptable to have the system actually fail, but be restartable within a short period of time.

Advantages:
- Fully fault tolerant.

Disadvantages:
- Cost
- Limited vendor selection
- Custom hardware and software

5.2 FAIL-SOFT/FAULT-TOLERANT SYSTEMS

Fail-soft or *fault-tolerant* systems incorporate elements of redundancy and/or modular component replacement to permit continued operation, sometimes in a degraded mode, when selected components fail.

Typically only the components that most commonly fail, and are most affordable, are made redundant—e.g., power supplies and disk controllers, but not the system bus—leaving some single point of failure components, but these are selected to be the most robust components in a system.

Thus, the trade-off is reduced cost and complexity for some possibility of system failure. This is an acceptable decision in many business environments, particularly in light of the extended MTBF seen in virtually all modern system and infrastructure components.

Particularly as component manufacturers have been addressing fault-tolerant issues, this solution also lends itself to use in either improving fault tolerance in vendor-built common platforms, e.g., Intel-based server solutions, or in fully generic servers built from off-the-shelf components.

Advantages:
- Significantly more inexpensive than fully non-stop system.
- Fault tolerant to common failures.
- Repair downtime may be scheduled for redundant component failures

Disadvantages:
- Down time is not eliminated.
- Integration of fault-tolerant components is OS, application, and vendor dependent.

5.3 FAILOVER SERVERS

Rather than assuring individual components are fault-tolerant, a common practice is to use a *failover server* (sometimes called a *warm spare*.) In the simplest of cases, this is literally a spare server—it's not used for anything else, and may not even be installed prior to failure of the in-service system. Typically, operator intervention is necessary to some degree in case of primary server failure, but this is almost always less than the time required to diagnose and repair the primary.

Since leaving an entire computer system idle awaiting the typically rare system failure is wasteful, there are permutations of this scheme, including:

- Making the servers fault-tolerant in their own right, thus reducing the probability of full failure before switching over to the spare.

- Use of the spare system to pick up processing tasks from the primary server as long as it's fully functional.
- Making use of the spare system to carry out unrelated missions that can be terminated immediately should the system be needed in its replacement role.
- Automated monitoring of the primary by the secondary to permit unattended switch-over in case of failure.

Advantages:
- Significantly more inexpensive than fully non-stop system.
- Much shorter downtime in case of failure.
- Fault tolerant to common failures.

Disadvantages:
- Down time is not eliminated.
- A fully functional replacement server must be maintained.
- Some degree of complexity is introduced to support switchover.
- Switchovers may require manual operator intervention.

5.4 CLUSTERING

Some hardware and operating systems are capable of *clustering* servers. In this model, separate servers, or *nodes*, nevertheless can combine their resources—CPU, communications, storage, etc.—to appear to the applications to be a single computer with the aggregate capabilities of its component nodes. Should any single server in the cluster fail, the remainder will simply pick up the workload and continue, albeit degraded by the loss of the capabilities of the failed node.

This may be considered taking failover servers to the extreme case. If fully functional, it gives the reliability of a fully non-stop system with the ability to make use of the redundant system's capabilities in day-to-day operation.

While clustered servers have been common in mini- and mainframe environments, and for some proprietary vendor systems, for years, they are just recently becoming commonly available on popular microprocessor-based systems such as Intel and Alpha machines running such operating systems as Windows and Linux. At this point in time, a clustered solution should be examined on a case-by-case basis to determine its applicability and maturity for the intended environment.

Advantages:
- Downtime is eliminated.
- More realized capability from hardware than a fully non-stop system.
- Failure mode is typically just degraded performance.

Disadvantages:
- Cost.
- Complexity.
- Restricted number of offerings from vendors.
- Maturity of some versions must be carefully examined.

5.5 ON-SITE REPAIR/REPLACEMENT

If it's not critical to be operational immediately—that is, some down-time, on the order of minutes to hours, is acceptable—the most inexpensive rapid-recovery scheme is to simply stock a full or partial set of replacement components on-site. The modular nature of modern systems makes replacement of a power supply, memory, or an entire motherboard a matter of a few minutes, provided technically competent personnel are on-site.

Vendors also offer rapid replacement and repair contracts, typically specified in terms of hours, e.g., guaranteed repair within 4 hours of notification. This may sometimes be coupled with storage of vendor components on-site (for larger sites), or use of customer-purchased stockpiled components.

Advantages:
- Often cheapest solution.
- Usually can provide recovery within specified time frame.

Disadvantages:
- Indeterminate downtime in case of component failure.

6. DATA STORAGE

Data storage is a complex arena, including traditional disk storage, various fault-tolerant models such a RAID arrays, and more esoteric considerations such as secondary or tertiary storage schemes and backup models. In this treatment, only *primary* storage will be discussed—data that is stored and accessed by the network users on an active basis.

Following sections discuss the various solutions that are available when considering fault-tolerant storage solutions.

6.1 NETWORK ATTACHED STORAGE

A critical decision must be made when deciding how to implement data storage in a fault-tolerant environment. Traditionally, especially in micro-based servers, disks and their controllers are components within the server, directly managed by the host operating system. This certainly guarantees the most rapid disk access—limited by controller and bus transfer rates—at the expense of exposing the disk storage subsystem to failure if/when the server itself fails.

In recent years, rapid delivery of significant increases in the processing power available to disk controller subsystems and in the bandwidth available to link components in the computer-room environment, coupled with more reliable network file sharing services have made Network Attached Storage (**NAS**) a viable solution. This simply refers to the ability to provide disk storage at a remove from the server(s) making use of it. The two common configurations for this are referred to as *Storage Access Networks* (**SAN**) and *NAS appliances*.

In such solutions, the disk(s) used for server storage actually reside in another component in the same network as the application server or servers.

In many ways both approaches are very attractive relative to traditional disk management, each providing the following advantages:

- Isolation of the data storage system from the server carrying out application-related tasks, allowing both to be optimized for their different tasks and usually resulting in a cheaper server requirement for the applications software.
- Protection the data subsystem from any failures that may strike the application server, since the data resides on a physically separate system.

- Substitution of different data transport mechanisms[1] and levels of equipment to allow for faster, bigger solutions when needed, without changes to the application server or its software.
- Combination of several fault tolerant methods, either just for data storage or the application server(s), or both, depending on business requirements, financial constraints, etc.

Despite the common advantages, there are strengths and weaknesses unique to each model.

6.1.1 NAS Appliances

In a NAS appliance (also referred to as *filesystem appliances*), data access is provided by a device that is a self-contained server in its own right, optimized for handling disk data transfer operations[2] and participating in the common network that links the servers and other network devices.

One particular advantage is that filesystem appliances need not use the native filesystem of any particular operating system. This allows both common support in a heterogeneous environment, and use of filesystem models more robust than are commonly encountered in general-purpose servers to enhance data integrity and recoverability.

Commonly, these incorporate fault-tolerant schemes such as internal RAID, battery-backed-up (BBU) memory, *mirroring*, and *journaling* filesystems that make failure recovery much more reliable and less dependent on external components or configurations than traditional or SAN solutions.

Advantages:
- Often cheapest detached storage entry-level solution
- Typically self-contained, very easily incorporated in existing networks.
- Communicates over standard communal LAN/WAN.
- Uses common network filesystem protocols.[3]

Disadvantages:
- Slower than either SAN or traditional disk access.
- Costs for larger or more complex configurations can rise rapidly.
- Many advanced features are separately priced additions.

6.1.2 Storage Access Networks

A SAN is a dedicated solution that makes the mechanics of *how* the disks and filesystems are managed transparent to the server applications, but is integrated into the server operating system[4]. Rather than communicating with the disks on a common internal bus[1] or even a

---

[1] 10-BaseT, 100-BaseT, or Gigabit Ethernet and fiber are all examples of how the underlying data transport mechanism can be upgraded. None require any cooperation from the applications involved.

[2] These either run embedded proprietary operating systems, such as Network Appliance's "Data ONTAP", or versions of familiar operating systems such as Linux or Windows 2000, usually stripped of extraneous services/software and optimized for filesystem management.

[3] e.g., NFS, CIFS (SMB).

[4] This is a loose description of a SAN. This may incorporate several different configurations, but usually makes use of custom operating system drivers, server disk controller hardware, and a high-speed dedicated network, external to the server itself, on which only communications between server disk controllers and the disks is carried.

dedicated internal data bus[2], SANs use an external very high-speed data channel, dedicated totally to disk data transfers, and custom controllers in the servers to exchange data with the SAN.

Typically, SAN solutions are more intrusive on the host operating system, since the OS must be able to communicate with the specialized controller cards, and often require management software to configure and control the SAN itself. They are also somewhat more complicated, and usually costly, than network attached storage in terms of number of discrete components and interfaces.

These drawbacks are compensated for by the ability to move huge amounts of data across a SAN very rapidly and safely. Particularly in enterprise environments with heavy data distribution, access, and backup requirements, a SAN can be the most cost-effective solution to these requirements.

Advantages:
- Very fast, capable of moving large amounts of data.

Disadvantages:
- Typically costly; entry level system are still expensive.
- More complex to configure and administer than NAS

6.2 RAID ARRAYS

RAID (*Redundant Array of Inexpensive Disks*) isn't a single solution. Rather, it denotes a family of approaches that express a common model: Use more than one disk drive to provide data redundancy in case of failure of any one drive or part of a drive.

6.2.1 Background

For quite a while in the mini and microcomputer environment, a disk held data as a self-contained unit, e.g., drive *C:* was the whole disk, and all the directories and files were rooted on that drive.

As drives got larger, it was possible to *partition* a disk—create logically separate structures that looked, to users and programs, like different hard drives, but in fact all resided on the same physical disk. Thus, now that same, but much larger, hard drive might appear to be three drives *C:*, *D:*, and *E:*, but in fact all reside on a single physical component. This allowed organization of data in smaller, logically related groups of information that were easier for people and software to manage.

Unfortunately, if that physical disk fails, all the information—whether on a single drive, or multiple partitions—is a risk of corruption or loss.

But going in the opposite direction, all data doesn't have to reside on a single physical disk. Rather, it's possible to use two or more disks that appear to the system users to be a single disk, and use various means to duplicate the information on the disks such that it can be accessed or rebuilt.

The simplest RAID solution was *mirroring*—simply guarantee that the operating system software or hardware always wrote exact copies of everything to two disks instead of just one. This was most wasteful of hardware—if you had two 10-megabyte (Mb) disks, you could store 20 Mb of data if they weren't mirrored, but only 10 Mb if they were—you lost half the total raw capacity!

---

[1] e.g., IDE controllers

[2] e.g., SCSI controllers

But this was more than made up for in some environments by the fact that if either disk failed, the other had a full and accurate copy of everything. Replace the bad drive, and the information from the good one can be used while it's being copied over to the new, empty drive. You're shortly back to having a duplicate of everything, with no loss and even without any down-time (if the hardware allows *hot-swapping*, or removal of the bad drive while the computer is still running).

Now, if something destroys *both* disks, you're dead again—and disks were expensive enough that losing 100% of the second drive was often hard to justify financially.

Other versions of RAID are simply attempts to get around this loss of data storage and vulnerability to two drives. All use two or more disks to spread the data across multiple physical components to reduce the possibility that enough can fail to destroy the information. Some use methods that scatter part of the data across the drives such that all data doesn't reside on any one disk, but mathematical reconstruction techniques allow recovery even if more than one disk fails at the same time. These various schemes are recognized by their levels, e.g., *RAID-0*, *RAID-1*, *RAID-2*, on up to *RAID-5*. There are a handful of vendor-specific schemes that aren't generally recognized, but have crept into the naming scheme.[1]

6.2.2 Implementation Choices

Today, virtually all implementations of RAID are either RAID-1 (*mirroring*) or RAID-5 (distributed parity). Use RAID arrays has proven so reliable that it's supported directly by many disk controller cards, and is the fundamental first-level tool underlying fault-tolerant systems. It's probably a safe statement that every fault-tolerant scheme will, and should, have some level of RAID as its starting point.

Advantages:
- Well-understood.
- Wide range of performance and capacity options.
- Strong industry support.
- Good to excellent data integrity and speed of access.

Disadvantages:
- Slower than dedicated single-disk solutions.
- More expensive than single-disk solutions.

6.3 REPLICATION

Replication is commonly used when speaking of databases, but also applies to replicated filesystems. Essentially, it's the ability to guarantee that automatic duplication of data is carried out between two or more locations in a timely, reliable and verifiable manner. Typically, it's spoken of when provided as a service of the operating system or an add-on package, but it may be provided as a part of an application (particularly when vendor-provided replication doesn't meet business requirements.)

There are a number of different restrictions and criteria applied to what is considered timely, reliable, etc.; when considering third-party solutions, it's necessary to carefully determine the criteria used by the vendor.

Replication may most often be considered in a clustering solution, when there is no NAS solution incorporated.

---

[1] A good, concise definition of RAID levels in common use can be found at http://www.enhance-tech.com/tech/raiddef.html

Advantages:
- Provides data copies to multiple locations
- Often requires no modification of applications
- Often an integral part of the operating system and/or database engine
- Can, and usually is, used in conjunction with RAID

Disadvantages:
- Wide range of criteria and implementation decisions across solutions
- Often less reliable than data storage solutions such as SAN or NAS appliance solutions

7. COMMUNICATIONS

Particularly as virtually all systems rely heavily on LAN and, increasingly, WAN connectivity, if considering the full spectrum of fault tolerance, addressing communications failures is a critical issue.

For the purposes of this paper, however, the focus is on server and data issues, placing deep consideration of communication issues beyond the intended scope. A brief visit to salient segments will be made, however.

7.1 NETWORKING

In individual servers and other active network elements such as NAS appliances or SAN devices, duplicate network interface cards (NICS) and the ability of the host software and/or operating system to dynamically switch between them in case of failure is obvious.

Other issues that must be addressed, and usually resolved via redundant components, are:

- Failure of active networking components (routers, hubs, switches)
- Failure of passive networking components (wiring runs, jack panel connections)

Determination must be made if the failover requires manual intervention, and the acceptable duration of any outage.

7.2 OTHER COMMUNICATIONS SERVICES

This would cover a gamut of traditional—serial communications, leased lines, DSL connections—and specialized communications such as wireless access points (WAPs), dedicated controller connections, etc.

8. POWER

The problem of providing clean, reliable power to all components in a fault-tolerant network is often only superficially addressed. Again, a deep examination of this topic is beyond the intended scope of this paper. However, the topic needs to be mentioned, simply because it is so often the Achilles heel of a fault-tolerant network architecture. Often the experience has been that each desktop and server has been provided with a UPS, only to have the entire network fail because a set of switches or a router were forgotten in a wiring closet.

Typically, uninterruptable power supplies need to be provided to servers and active network components. Often today, these UPS devices are network-aware in their own right, permitting a fair degree of monitoring and notification.

But external power failure is only one consideration. Power supplies are one of the most common component failures. Fault-tolerant devices themselves virtually always require dual power supplies, even if the devices themselves are duplicated.

9. CONCLUSIONS

High-availability systems and networks can be constructed from a wide variety of possible solutions, providing an equally wide range of fault tolerance capabilities predicated on business needs and financial constraints. This document has enumerated the choices that are generally in use and commercially available.

Construction of a fault tolerant system or network requires the consideration of these solutions, their interaction, and their applicability individually and collectively to the problems at hand.

==END OF DOCUMENT