TITLE

Constructing Mission Critical Solutions using Hewlett-Packard's New High-end HP-UX Server.

Ken Pomaranski
Hewlett-Packard Company
8000 Foothills Blvd. Roseville, CA
Phone: (916) 748-2436
Fax: (916) 748-2335
Email: ken_pomaranski@hp.com

ABSTRACT

The next generation HP high end server will offer unprecedented availability through strong Single System High Availability (SSHA), leadership multi-system High Availability (HA) with MCServiceguard, and world-class fault and event management capability. The growth and maturing of the Storage area network (SAN) architecture is also a key enabler. This paper describes how to put these technologies together to create a maximum uptime system level solution.

A high availability solution is more than just a single server with hot-swappable fans running an important business critical or mission critical application. Every part of the system solution must be 'tuned' for high availability. This not only includes the server, but the network, peripherals, database, applications, operating system, service, support, site infrastructure, and IT processes.

This paper will examine the entire high availability value chain from solution design, through installation and configuration, to ongoing customer care and solution maintenance. The actual downtime event data is used as the basis for determining the relative priorities of the problems that must be solved in engineering a solution.

Each component of the system solution will be broken down, and its contribution to overall solution uptime studied. The methodology will be to look at end-to-end availability by breaking it down into downtime causes for each section of the HA value chain, then to fully describe the system solutions that address each of these areas. The aggregate system level solution and the ‘HA pyramid’ will then be presented.

INTRODUCTION

Most installations of High Availability solutions fail to take into account the main causes of downtime when constructing a grounds-up solution. Therefore, solutions that ‘on paper’ appear to deliver high levels of availability are inadequate in practice.

This is because it is much easier to install and configure redundant hardware to address high availability than it is to create rock solid data center and IT processes. Hardware is very concrete and therefore

relatively easy to understand. Another reason is that conventional wisdom dictates that hardware problems are the predominant failure mode in high availability solutions. Create a solution with fully redundant hardware and presto! all problems are solved.

In fact, it has been shown across Hewlett-Packard's (HP) customer base (and across all computer users as documented in industry research about this topic) that achieved availability varies widely across the industry with roughly the same Hardware (HW) configurations.

Why is this? Obviously, there is more to delivering High Availability than just throwing redundant hardware at the problem! This is, however, a very necessary and important part of the solution. It must be done flawlessly to build a strong enough base to deliver the highest levels of availability.

This paper will examine the end-to-end system and processes necessary to achieve "three-, four- and five-nines" availability solutions. (99.9%, 99.99%, and 99.999% solution uptime.)

AVAILABILITY LEVELS

Availability Level	*Representative Applications*	*Availability required*	*Downtime per year*
Reliable system	print servers	99% - 99.5%	44 hrs -> 87 hrs
Resilient system	batch processing	99.5% - 99.9%	8 hrs -> 44 hrs
Highly available system	backend database servers	99.9% - 99.99%	52 min -> 8 hrs
continuously available system	air traffic control	99.999%+	5 minutes or less

WHAT IS HIGH AVAILABILITY?

Two definitions currently used are:

(1) Hardware and software systems designed to minimize planned and unplanned downtime.

(2) "Hardware and software systems designed to protect against component and system level failures and when a fault or failure does occur, data is not lost and the system can recover in a reasonable amount of time" -- Source: IDC

Both these definitions are somewhat inadequate, as they don't tell the whole story. A better definition might be:

"Hardware systems, software systems, IT/support processes, and data center infrastructure designed to minimize downtime events and their associated recovery times."

WHAT IS DOWNTIME?

The time that a customer's system is unavailable to do useful work, usually measured in hours / year (or converted to % uptime). If a failure can be 'masked' (i.e. the customer's system is still running

during the entire failure event, including repair) the failure and its associated repair times do NOT count toward downtime.

Downtime per year is measured by the following formula:

(# of downtime events per yr.) x *(mean time to recover from events)* = downtime

The thrust of HA is evident from that formula:

Increase system availability by either limiting the amount of downtime events per year, or by reducing the impact to the customer of these downtime events, i.e. reducing mean time to repair (MTTR).

Obviously, the best way to reduce downtime based on the above formula is to REDUCE THE NUMBER OF DOWNTIME EVENTS! This is because for each downtime event, you must 'pay' the time to recover from that event, and this payment is expensive!

To give an idea of just how expensive, examine the components that make up an unplanned downtime event:

Fault / Event occurs	0 minutes
Crash dump / user notified	5 - 20 minutes
Response center contacted	1 - 60 minutes
CE response time	120 - 240 minutes
Diagnose time	10 - 240 minutes
Repair time	10 - 180 minutes
Retest / verify fix	5 - 20 minutes
Reboot time	12 - 30 minutes
Database recovery	5 - 480 minutes
Application restart	0.5 - 10 minutes
TOTAL	3 - 20+ hours

Note that it is impossible to meet the highest levels of availability even if only ONE unplanned event occurs! (Of course, some optimization can be done to lower the above numbers further. For instance, doing diagnose and CE response time in parallel or having spare parts and possibly a CE on-site can greatly improve the downtime. More will be discussed on this later.) But even with these optimizations made, 4 and 5 nines availability levels are out of reach.

This is also the case for planned events:

Quiesce system	0 minutes
Repair time	10 - 60 minutes
Retest / verify fix	5 - 20 minutes
Reboot time	12 - 30 minutes
Clean database recovery	5 - 10 minutes
Application restart	0.5 - 10 minutes
TOTAL	.6 - 2+ hours

Even in the best case, the entire four- and five-nines budget is eaten up!

So what can be done? To answer this question, it is necessary to explore the causes of downtime, and come up with techniques to either:

(1) Eliminate these causes.
(2) Make the system level recovery time as short as possible.

From here forward, each downtime cause will be called an ‘event’ and the downtime associated with each event will be referred to as the ‘cost’ of the event.

MODEL PRESENTATION AND WALKTHROUGH

There are 2 basic configurations to examine:

1. Single-system High Availability (SSHA)
2. Multi-system High Availability (high availability clusters).

The downtime events that effect each of these configurations are:

UNPLANNED EVENTS
Server hardware failures
Networking errors
Mass Storage failures
Operating System (OS) failures
Application / database faults
Site power interruption
User / operator error

PLANNED EVENTS:
OS patches
Kernel tunables / SW configuration changes
HW upgrades & configuration changes
Database reconfiguration

What causes each of these downtime events? What is the contribution of each to the overall availability of a system solution?

The following model attempts to quantify the various effects on system availability caused by the specific downtime events listed above. The model is somewhat complex, but the understanding of it is extremely important to gaining insight on how to build four-nines or five-nines configurations.

The rest of this paper will be dedicated to explaining the model, and more importantly, detailing the implications.

		% of total	standard events per year	optimize factor	opt. events per year	MTTR	Weighted MTTR	Opt MTTR	Weight MTTR
	UNPLANNED EVENTS								
	Server hardware failures	20%	0.70	20%	0.56	4.34	3.04	3.34	1.87
	Networking errors	15%	0.525	80%	0.105	3.9	2.03	2.88	0.30
	Mass Storage faults	4%	0.14	80%	0.028	4.34	0.61	3.34	0.09
	Operating system faults	10%	0.35	20%	0.28	1.84	0.64	1.84	0.52
	Application faults	10%	0.35	70%	0.105	1.84	0.64	1.84	0.19
	Database faults	10%	0.35	70%	0.105	1.84	0.64	1.84	0.19
	Site power interruption	8%	0.28	95%	0.014	1.84	0.52	1.84	0.03
	Operator error	23%	0.805	95%	0.04025	3.34	2.69	3.34	0.13
	TOTAL	100%	3.5		1.24		10.82		3.33
	PLANNED EVENTS								
	Planned OS patches / year	57%	4	75%	1	1.01	4.03		1.01
	Planned OS kernel configuration changes	14%	1	75%	0.25	0.88	0.88		0.22
	Planned HW downtime for IO upgrade	7%	0.5	95%	0.025	2.38	1.19		0.06
	Planned HW downtime for for cell upgrades	7%	0.5	95%	0.025	2.38	1.19		0.06
	Planned HW downtime for other HW updates	7%	0.5	75%	0.125	2.38	1.19		0.30
	Database reconfiguration	7%	0.5	75%	0.125	0.88	0.44		0.11
	TOTAL	100%	7		1.55		8.9		1.75
	HW RAW AFR	200%							
	HW resiliency percentage	65%							
	HW MTTR (includes diagnose, and test)	1.5	hour						
	HW MTTR reduction due to EMS monitors	1.00	hour						
	CE response time	2	hours						
	Multi-node switchover time	0.008	hours	0.5	min				
	Multi-node switchover failure percentage	1%							
	Window of vulnerability (N-1 systems)	4	hours						
	% patches that are bad	15%							
	PDC boot time	0.08	hr	5	min				
	OS dirty reboot time	0.33	hr	20	min				
	OS Clean reboot time	0.20	hr	12	min				
	OS patch installation time	0.50	hr	30	min				
	DB dirty recovery time (normal database)	0.42	hr	25	min				
	DB clean recovery time (normal DB)	0.08	hr	5	min				
	DB dirty recovery time (fast DB recovery)	0.01	hr	0.5	min				
	Application restart time	0.01	hr	0.5	min				
	Crash dump / SW debug time	1.00	hr						

Expected availability, standard configuration, standard data center practices

	Single System		Generic DB, HA cluster		HA cluster w/ fast DB			
	AFR	downtime	AFR	downtime	AFR	downtime		
		hours		hours		*minutes*		
Server failures	70.00%	3.04	70.00%	0.331	70.00%	2.730		
Failures while cluster is 'split'	0.00%	0	0.03%	0.001	0.03%	0.077		
Network failures	52.50%	2.03	10.50%	0.420	10.50%	25.200		
Mass storage failures	14.00%	0.61	2.80%	0.112	2.80%	6.720		
OS failures	35.00%	0.64	35.00%	0.166	35.00%	1.365		
Database / app failures	70.00%	1.29	70.00%	0.331	70.00%	2.730		
Site issues / user error	108.50%	3.21	108.50%	0.514	108.50%	4.232		
OS patches	400.00%	4.025	400.00%	0.560	400.00%	33.600		
SW reconfiguration	150.00%	1.3125	150.00%	0.210	150.00%	12.600		
HW upgrades	150.00%	3.5625	150.00%	0.210	150.00%	12.600		
Downtime hours / year		19.72	hrs	2.86	hrs	101.85	minutes	
Availability		99.7750%		99.9674%		99.9806%		

Expected availability, well controlled environment, optimized configuration

	Single System		Generic DB, HA cluster		HA cluster w/ fast DB			
	AFR	downtime	AFR	downtime	AFR	downtime		
		hours		hours		*minutes*		
Server failures	56.00%	1.87	56.00%	0.265	56.00%	2.184		
Failures while cluster is 'split'	0.00%	0	0.03%	0.001	0.03%	0.077		
Network failures	10.50%	0.30	0.11%	0.004	0.11%	0.252		
Mass storage failures	2.80%	0.09	0.03%	0.001	0.03%	0.067		
OS failures	28.00%	0.52	28.00%	0.133	2.80%	0.109		
Database / app failures	21.00%	0.39	21.00%	0.099	2.10%	0.082		
Site issues / user error	5.43%	0.16	5.43%	0.026	2.71%	0.106		
OS patches	100.00%	1.006	100.00%	0.140	50.00%	1.950		
SW reconfiguration	37.50%	0.328125	37.50%	0.053	3.75%	0.146		
HW upgrades	17.50%	0.416	17.50%	0.025	1.75%	0.068		
Downtime hours / year		5.08	hrs	0.75	hrs	5.04	minutes	
Availability		99.9421%		99.9915%		99.9990%		

There are two sets of data that feed into the six system models. The two sets of data are:

1. Downtime events and frequency of each averaged across a large customer base. The event frequency is given for both a ‘standard’ data center and a data center with well controlled processes which will be referred to from here forward as an ‘HA data center’.

2. ‘Recovery times’. These various recovery times are used to estimate the average downtime incurred per event. The average downtime is usually referred to as the ‘mean time to repair’ or MTTR. Again, the recovery time per event is also known as its ‘cost’.

The six system models are:

1. Single-system availability, ‘standard’ data center

2. Multi-system availability, assuming an HA cluster and a ‘generic’ application, standard data center.

3. Multi-system availability, assuming an HA cluster and an ‘HA database’, standard data center.

4. Single-system availability, HA data center and processes.

5. Multi-system availability, HA cluster, generic application, HA data center and processes.

6. Multi-system availability, HA cluster, ‘HA database’, HA data center.

COLUMN EXPLANATIONS

Events per year, optimize factor, optimized events per year

The event frequencies given in the model for ‘standard events per year’ are an aggregate of many sources. These sources include:

1. HP field surveys
2. HP field return data
3. HP MTTR data
4. Publicly available information
5. Dataquest research
6. Gartner Group research

Obviously, a particular site can experience a different distribution across the events. For example, user error contribution has been shown to vary widely across the computer user base. It can range from between 5% to over 40%!

However, the solutions presented are applicable to ALL sites. This is because everything must be controlled to achieve high availability levels.

Therefore, it is important for a site that truly needs High Availability to use a product such as ‘HA meter’ from HP to keep track of what exactly is causing system downtime. ‘HA meter’ is a part of High Availability Observatory (HAO). A product such as this will show what areas are under control (have already been optimized) and which downtime event areas still need improvements.

Also note that the number of ‘server failures’ refers to a single system, NOT a high availability cluster.

There are two columns labeled ‘events per year’; one is the ‘standard events per year’, the other is the ‘optimal events per year’. The latter number is considered to be what *can* be obtained if certain architectures or processes are put into place. For example:

1. Careful design of the system architecture (planning).

2. Data center designed for HA
3. Careful change control policies
4. Careful management of access to critical machines (from both inside and out.)

These areas make the difference between a 'resilient site' and a 'highly available site'.

It is important to note that each area has a different sensitivity to improvements. For instance, user error and site problems can be nearly eliminated, whereas server HW and SW problems are largely a function of the design and manufacturing processes of the HW vendor and the fact that hardware does fail. Large expenditures over and above implementing the correct architecture for the application will not yield the desired results.

The 'optimization factor' is a measure of how designs and processes at a particular installation can reduce the occurrence of downtime events. Note that there is a practical limit to what can done at an installation to reduce single server failures (HW or SW). These elements are inherent to the robustness of the server. The main areas that can be influenced are in the 'infrastructure' areas.

MTTR explanations

The MTTR column is the average time it takes to get a system back online after experiencing the particular 'event'. The weighted MTTR is the standard events per year multiplied by the MTTR. The Opt(imized) MTTR is the mean time to repair after process / design improvements are put into place. The last column, 'Weight MTTR', is the Optimized events per year multiplied by the Optimized AFR.

Adding the Weighted MTTRs yields the average downtime per year for a single system.

ROW EXPLANATIONS

Event types

As noted before, downtime events are categorized as 'unplanned' or 'planned'.

Unplanned downtime events occur without warning. They can be devastating because of their nature. These events can occur at any time during the course of business.

Planned events occur at prescheduled times. They still count against downtime because the system is unavailable during this time.

Hardware failure rate

The raw 'annualized failure rate' (AFR) of hardware is quoted as a percentage. The 200% number means that, on average, every system in a large population of systems will experience 2 HW faults per year. The value of 200% is the RAW failure rate, not the failure rate that actually causes system downtime. Many faults are 'hidden' by High

Availability features, making the ‘apparent failure rate’ much less. This raw number will scale as the system grows (or shrinks) in size.

The ‘apparent failure rate’ is modeled by the ‘resiliency percentage’. The apparent failure rate is the RAW AFR multiplied by the resiliency percentage. This percentage will vary based upon the number and type of Single System High Availability (SSHA) features implemented in the server.

Hardware MTTR
Hardware MTTR is the amount of elapsed time between the CE arriving on the site and the boot processes started.

EMS monitors
The EMS monitor row refers to the amount of time reduction in MTTR due to the presence of the full fault management suite (including EMS monitors).

Customer engineer (CE) response time
This is the maximum time from the response center call to the CE arriving on site with the correct solution to the problem.

Multi-node switchover
This is the switchover time (from failed node to a running node). Also models ‘switchover failures’ due to configuration or other problems.

Normal database switchover
This is the fail-over time of a standard database

HA database fail-over
This is the fail-over time for an HA designed database.

SUMMARY OF MODELS & IMPLICATIONS / OBTAINING HIGH AVAILABILITY

High Availability can be obtained without the expenditure and manageability problems inherent with fault tolerant hardware.

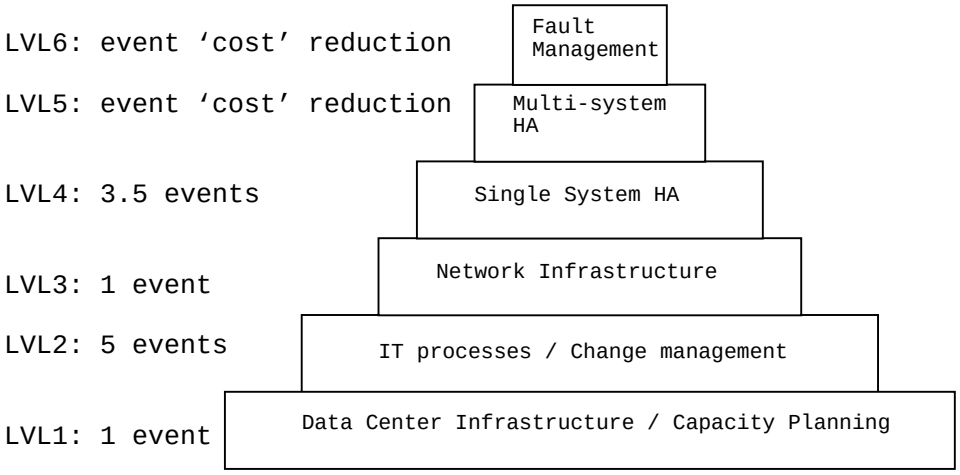
However, the highest levels of availability (four- and five-nines) cannot be achieved without implementing measures that both limit the number of downtime events per year AND reduce the impact of each downtime event.

As the model shows, the total number of downtime events cannot exceed three per year to make these levels of availability, and the time per event needs to be less that five minutes per event.

So, how can these goals be achieved?

The best way to choose the right solution and obtain these levels of availability is to sort the events detailed above into the ‘HA Pyramid’. The HA Pyramid is divided into six levels. Each level in the pyramid is dependent on the levels below it:

Pyramid level & events / year



As shown in the model, focusing only on redundancy (LVL4 & LVL5 in the pyramid) and ignoring the other causes of system downtime, makes the highest levels of availability unobtainable. This is a common problem with many customer installations.

Obtaining the highest levels of availability requires addressing all six levels of the pyramid.

The last part of this paper will address the HA pyramid specifically and demonstrate how to design architectures and processes that result in four- and five-nines configurations.

LVL1: Data Center Infrastructure / Capacity Planning

Data Center Infrastructure

This is the base of the pyramid! Data centers need to be built with redundant input power sources. Site uninterruptible power supply (UPS) is also a good idea. A UPS per system or set of systems is not a good solution in the long run.

Redundancy also needs to be built into the power routing infrastructure. The best solution is to have two power sources available to each device in the data center. Therefore, if maintenance needs to occur on one ‘grid’, the other is still available to provide power.

The best servers and mass storage devices will provide the means to connect to both power grids simultaneously and eliminate the need for a ‘switchover’ during a power event.

Cooling is another consideration. The cooling infrastructure needs to be developed to ensure cooling availability to the data center during maintenance. Capacity planning is crucial. It is important to note that new models of servers are NOT getting cooler in the long run!

New data centers should be developed as per specifications developed by the Uptime Institute or other data center architecture experts.

Another possibility to explore is ‘renting’ data center space from a server hosting company such as Exodus Communications, or from an Application Service Provider (ASP) such as Quest communications or US Internetworking. (But, this may present additional problems which need to be addressed. These problems have to do with the physical separation of a business and the data center that runs the business).

Capacity Planning

Some of the biggest causes of downtime events are software and hardware upgrades. This is also an area that a server owner has quite a bit of control over.

The two components that need to be addressed are:

1. Data center capacity
2. Individual server capacity

Besides data center capacity at the highest level (can the data center fit the required number of servers now and 10+ years out?), the data center floor space must be managed.

For individual servers, scalability is important. Online scalability is even better. Scaling can be broken into two types: (1) Adding a node (or a set of nodes) to an HA cluster, or (2) Adding capacity to an individual server.

Adding a node to an HA cluster is desirable, but may not be feasible due to floor space limitations or the ability / inability of the application to be distributed.

Adding capacity to an individual server entails adding CPUs, memory or I/O to an existing system.

Both types of capacity additions are made easy with the new SuperDome server. More CPU capacity will be added online with the instant capacity on demand (iCOD) functionality and more I/O capacity can be added with online PCI card addition. Also, new system partitions can be created (or added to) if more are required. Each SuperDome will support up to 64 CPUs (128 or more in a later release). This will help in capacity planning.

LVL2: IT Processes / Change Management

Nearly half of all downtime events can be eliminated through improvements in IT Processes and change management! Stability is important. It has been shown time and time again, through much research, that systems that are left alone experience much less downtime than those systems that are constantly being ‘tuned’. (The old adage: ‘If it ain’t broke, don’t fix it!’ comes to mind).

Number one rule: MINIMIZE THE NUMBER OF CHANGES

This requires careful advanced planning and an understanding of the application and the subsequent load placed on the system. Plan for future capacity upgrades.

Obviously, configuration changes will be necessary. The system vendor has control over making some of these easier, but overall, the customer is in ultimate control of making the improvements necessary to gain the desired availability levels.

HP / Database vendor provided ‘help’:

1) Massive patch reduction / patch quality effort going on inside of HP.
2) Reduction in number of kernel tunables. Making those tunable that remain dynamic.
3) Online addition of PCI cards
4) Online addition of CPUs
5) Configuration tracker (provided by the High Availability Observatory, HAO)
6) Online database reconfiguration
7) Online archiving

Customer improvements necessary:

1) Automate the change process wherever possible. (use scripts, or job scheduling mechanisms).
2) Document processes for EVERYTHING. Don’t assume that a well-trained operator will be performing all operations.
3) Standardize configurations! Have each node in a cluster use the same configuration.
4) Create accountability for downtime. Reward for meeting goals. Inspire changes in areas that don’t meet the goals.
5) MEASURE downtime and its causes, fine-tune processes to eliminate these events.
6) CREATE A CHANGE MANAGEMENT PROCESS

This last point is by far the most important, and will be detailed further.

Change Management

As stated earlier, some of the biggest causes of downtime events are software upgrades, hardware upgrades, and configuration changes. Many high visibility outages (i.e., those documented in the Wall Street Journal) were caused by lapses in this area.

A successful change management process is measured by the production environment uptime experienced during a change to the configuration. Success of this process should NOT be measured by the number of changes made, or by the elapsed time between the planning of the change and the implementation of the change. The QUALITY of the change is by far the most important element.

To meet this goal, any change management process should encompass the following elements:

1) Minimize the number of changes

2) Keep track of the configuration of all servers. Include hardware / firmware versions, and software versions.
3) Implement a change request process.
4) Coordinate changes, if the risk is low.
5) Schedule changes in advance.
6) Be cognizant of ALL customers who may be affected by an outage, and communicate change and ramifications to all affected.
7) Document all changes. Update the current configuration document for each server.
8) Build and test changes on an offline system! (or system partition, in the case of SuperDome).
9) If possible, implement a distributed application that is scalable over a number of nodes. This eliminates dependence on a single node and allows easier change management.
10) Avoid SW patches unless it is necessary to fix a blocking bug. (However, it may be necessary to update once every two years or so to keep the system at a supported configuration.)

LVL3: Network Infrastructure

The explosion of the internet and the resulting shift in the way business is being done is making the availability of the network one of the most critical elements in delivering high availability configurations.

Network backbones must be resilient. If a network switch fails, there are still redundant paths to connect to the remaining switches, i.e. dual paths from each server connecting to the backbone. Move network intelligence closer to the wiring closet and even into the workgroup.

There should be no single point of failure on the switch itself. A failed port or complete failure of a switch should not bring down the entire network. Also, network diagnostics and fault management software are vital.

Consistency is key. Use the same brand and type of adapter cards, switches, router cards, etc. These steps make it easy to stock spare parts and remove interoperability as a cause of downtime events.

LVL4: Single System High Availability (SSHA)

Now the base has been built! Continuing to build, it is now time to examine SSHA. This element is very important because it not only determines the amount of downtime per year for a single system for all SW & HW events, but it also determines the number of switchover events per year for multi-node HA clusters caused by these events. SSHA is best thought of as the ‘quality level’ of an individual box and its attached peripherals.

Single system HA typically includes the following events:

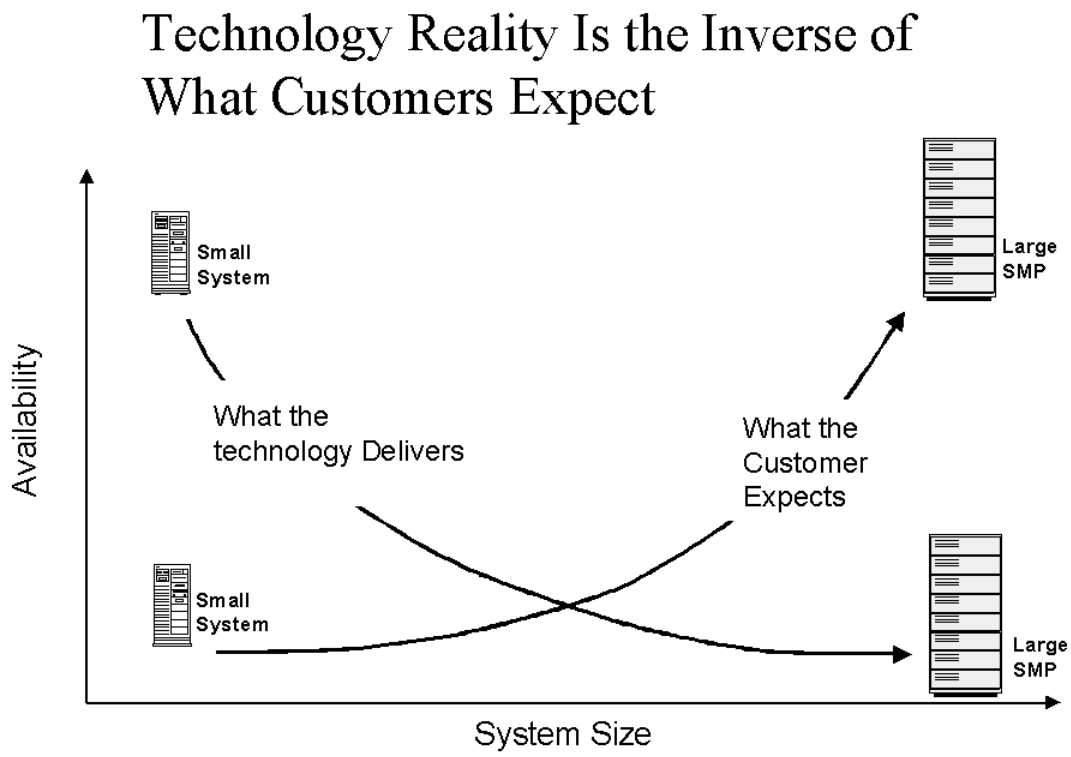
- OS and networking hangs and panics
- SPU hardware problems
- ‘Minor’ scheduled configuration changes (OS patches, FW updates, memory upgrades, I/O card addition)
- Peripheral problems
- File system corruption and recovery

Items that are NOT included under SSHA (These are included under 'IT process / Change management'):

- RDBMS backup/maintenance
- Major OS upgrades
- Scheduled system shutdowns for preventative maintenance
- Operator or user errors
- Application hangs or aborts
- Major SPU upgrade or replacement

To meet the highest levels of availability, it is imperative to minimize the number of yearly switchover events. Even if an event has a relatively small 'cost', several events can add up to devourer an entire four- or five-nines budget!

As stated before, SSHA is thought of as the quality of the server box inside a solution. Unfortunately, quality tends to go down with system complexity:



This fact makes it necessary to implement resiliency features in the box to compensate for the extra hardware necessary to deliver the desired performance levels. Some SuperDome features are:

Online addition / replacement of PCI cards
Add and replace PCI cards online.

DRAM fault tolerance
The SuperDome memory system is architected in such a way as to allow system tolerance to any single DRAM (dynamic random access memory chip) failure per DIMM set. No multi-bit error physically located in the same chip can bring a system partition down.

N+1 / Hot-swap front end power supplies
The AC to 48V front-end converters are N+1 and hot-swappable.

Dual AC Power feed
Each SuperDome cabinet (32w per) can connect to two completely independent AC sources.

N+1 / Hot-swap fans
All system fans are N+1 and hot-swappable.

Redundant / Hot-swap DC/DC converters
The DC/DC converters that supply power to the system backplane board are N+1 and hot-swappable.

Hot-swap guardian service processor board
The guardian service processor board (performs partition boot, initial partition configuration, power control for all entities, consoles for each partition, amongst other functions) can be replaced without interfering with system operation.

I/O Link resiliency
CPU controller to I/O interface is protected by full single bit stuck-at fault detection and correction.

Fabric resiliency
Cell-to-cell (each cell holds 4 CPUs and their associated memory) communication has multi-bit error detection and single bit "stuck-at" detection and correction. Cabinet-to-cabinet communication (through the high-speed link) has full retries on a garbled packet.

Diagnostic monitors & Fault notification
There is full system monitoring of all system ‘events’. More on this will be discussed in the fault management portion of this paper.

SW quality initiatives
There are significant quality improvements over the already good quality 11.0 HP-UX release. Also, a patch reduction program is being implemented.

CPU de-allocation
The system has the ability to automatically de-allocate degrading CPUs online. The system can also swap in a ‘spare’ CPU if one is available.

iCOD (Instant Capacity On Demand)
If a CPU is required, it will come online to be used by the operating system.

System Partitions
The architecture of the system allows the user to group sets of CPUs and IO together to form ‘system partitions’. Each partition is physically distinct from every other partition, except for some

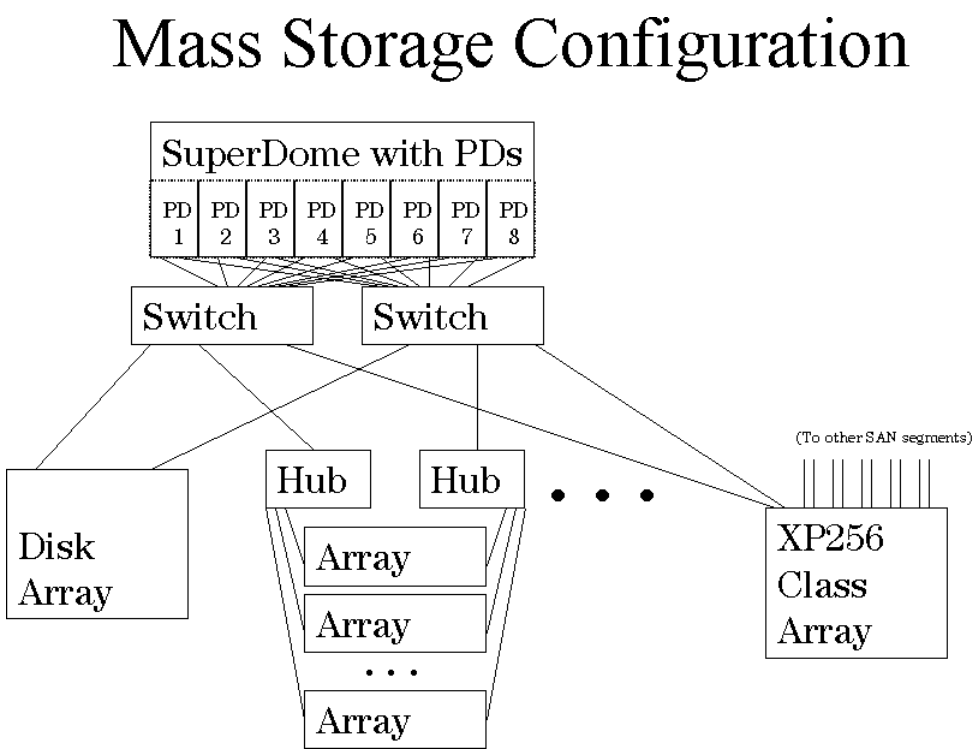
extremely low failure rate items (system clock and cabinet power control). Each partition can run different versions of the HP-UX operating system, and in later versions, possibly run NT, HP-UX, MPE, and LINUX in different partition definitions (PDs) in the same box.

This feature set results in a 32-way SuperDome having the same user visible failure rate as many of the 8-way servers in the marketplace today (and at much greater performance levels and system configurability).

In order to achieve these levels of availability, the system must be set up in such a way as to reduce the single points of failure per system partition. This includes multiple cells per partition, dual path I/O (for network, mass storage, and core I/O), at least eight DIMMS per cell, and extra CPUs for online addition in case of a CPU de-allocation, or an iCOD based upgrade. More information is available in the SuperDome system configuration guide and configuration white paper.

Mass Storage

RAID disk arrays (RAID level 5) and disk mirroring (RAID 1) are a must. Data must be protected at nearly any cost in high availability applications. There must be dual paths from a single system to any storage device the server needs to access. New implementations should use SAN networks. A well designed configuration may look like this:



LVL5: Multi System High Availability

It is important to understand that multi-system (or multi-node) availability, like all forms of redundancy, serves to make downtime events less severe. Redundancy does not prevent the initial failure event from occurring. In other words, multi-system HA reduces the 'cost' of a downtime event.

This reduction is significant. A typical downtime event for a single system consists of the following elements:

Fault / Event occurs	0 minutes
Crash dump / user notified	5 - 20 minutes
Response center contacted	1 - 60 minutes
CE response time	120 - 240 minutes
Diagnose time	10 - 240 minutes
Repair time	10 - 180 minutes
Retest / verify fix	5 - 20 minutes
Reboot time	12 - 30 minutes
Database recovery	5 - 480 minutes
Application restart	0.5 - 10 minutes
TOTAL	3 - 20+ hours

In multi-system HA, these elements still occur on the failed system, but the overall solution keeps running after the application switches over to a secondary node. The only downtime visible to the user is the 'switchover time'.

Switchover time is measured as the elapsed time between the application becoming unavailable on one node, to the application becoming available again on a secondary node.

Switchover time consists of:

1. Fault detection time & HW switchover
2. Database recovery time on secondary node
3. Application restart time on secondary node

The first element is provided by Serviceguard, Wolfpack, or in a few cases, by the application itself. The second two elements are a function of the particular database and application, both of which are controlled by the end user through their choice of software and software maintenance procedures. Note that some 'stacks' have been tested by HP and therefore are more likely to meet the expected availability levels. These 'Sweet-spot solution configurations' should be given priority for consideration when designing a new solution from the ground up.

Switchover times range from about 25 minutes down to less than a minute, depending on the database / application stack. Databases with fast restart capability, and those that can simultaneously run on multiple systems in a cluster (like Oracle Parallel Server, for instance), will perform in the faster-end of the range.

Hence, each downtime event can 'cost' between 1 and 25 minutes per event.

Remember that Four-nines availability amounts to about 50 minutes per year, and Five-nines availability amounts to about 5 minutes per year.

This means that the total number of downtime events, planned and unplanned, must be limited to no more than three or four per year to reach these highest levels of availability.

Also, care must be taken to ensure that single points of failure are eliminated between nodes in an HA cluster. A failure that brings down multiple nodes in a cluster will destroy any chance of making four- and five-nines. These types of faults include disasters, site power issues, cluster configuration problems, user error (hence the importance of the base of the pyramid!), and switchover failures.

Switchover failures are rare, and can be mitigated by stress testing the production configuration before bringing online.

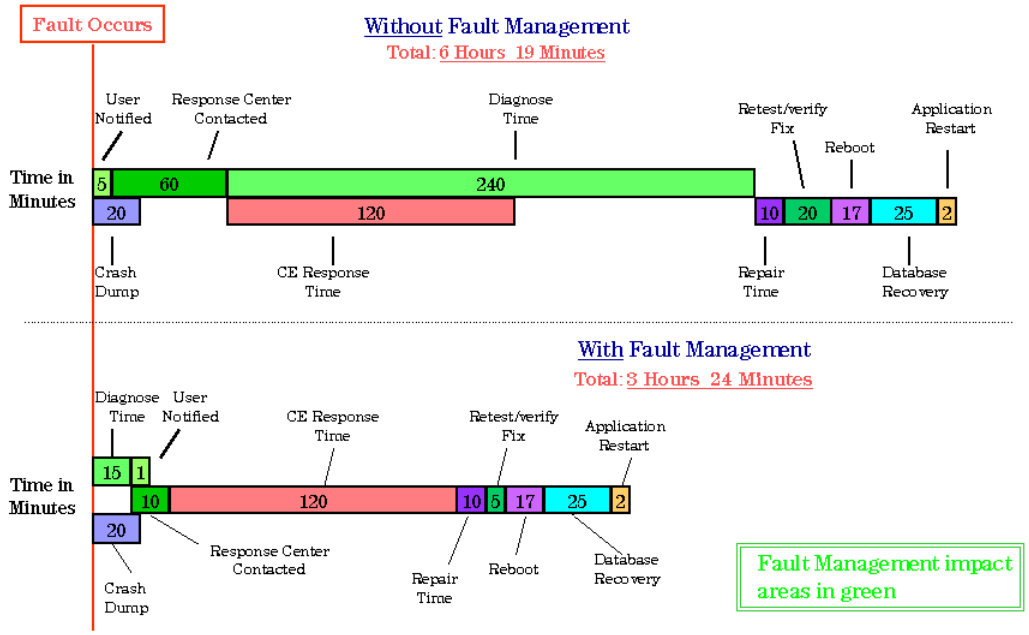
Mass Storage

Requirements are similar to those presented under single system HA. The only additional architecture element is to make sure that each node in an HA cluster has access to the same data.

LVL6: Fault Management

The goal of fault management is to change problem reporting and diagnosis from a reactive process to a proactive process. The typical fault tree with all events occurring serially has been presented earlier. What does the fault recovery process look like with proactive fault management?

Fault Resolution Comparison



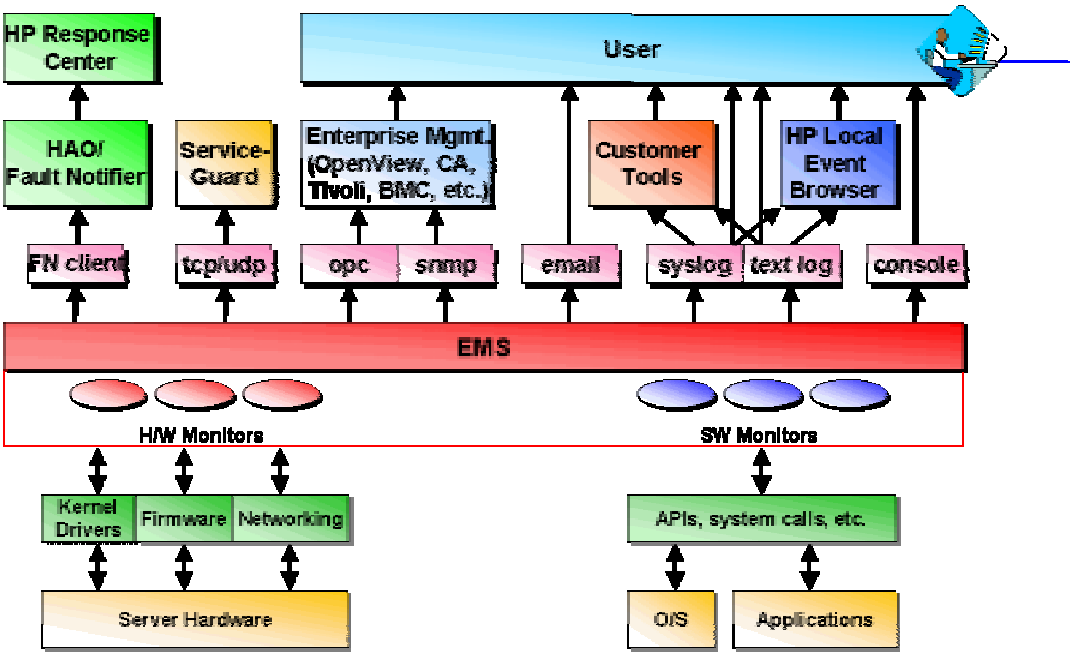
This time can be compressed further with on-site support, spare parts on-site, and fast database recovery.

Note that this ‘recovery time’ (time from fault occurrence to application up and running) is still important for multi-system HA. The reason is that the application is ‘exposed’ while one of the servers in the HA cluster is down. In fact, in order to statistically meet an average of five-nines availability over a large customer base, the ‘window’ of exposure to a second fault needs to be less than four hours.

Proactive fault management includes predictive analysis of system recoverable errors and reporting of cause-action to remedy the situation. It also includes real-time analysis of chassis code streams, automatic analysis of error log dumps. The ultimate goal is to eliminate the need for offline diagnostics for problem isolation.

The fault management architecture that delivers the desired level of ‘event time reduction’ is detailed on the next page:

Fault Management Software Stack



CONCLUSION

Four-nines and five-nines levels of availability can only be obtained by minimizing both the number of downtime events (primary consideration), and the 'costs' of downtime events.

Minimizing the number of events requires knowing the cause of downtime events. The most significant causes of downtime are NOT in areas that conventional wisdom dictates. Most downtime events can be avoided by hardening the *infrastructure* in which the server and its associated peripherals reside.

Once these elements are dealt with, hardware and software systems will then deliver the desired availability levels.