The Layered Availability Model

A Systematic Approach to Building High Availability

Prepared by Pete Gulotta

Presentation # 107

2597 Charlestown Road
Phoenixville, PA 19460
Phone: 610-933-0922
peteg1@erols.com

The Layered Availability Model

A Systematic Approach to Building High Availability

Introduction

Business managers today are more aware of the availability of their computer systems than ever before. When systems that support mission critical applications fail, the business generally loses revenue in one form or another. In the case of a failed factory floor system, goods can not be produced or shipped. In the case of a failed development environment, dozens of highly paid software designers sit idle. In any case, a failed system can affect a business's bottom line. High availability, or protection from such failures, is now a requirement in most mission critical systems.

Building a highly available system is all about assessing and accepting risk. Once the probability of failure has been assessed for all elements of a system, you must evaluate your willingness to accept such risk. There must then be a contingency for each risk too high to accept. Each contingency will have its own lower probability of failure. Ultimately, every system will have some probability of failure, however small, that the owner assumes. The goal of the systems designer is to minimize this risk.

This paper is written for systems designers and IT managers responsible for building highly available computer environments. My intention is to describe a systematic approach to designing and building such environments and a methodology for customizing that approach for a given set of business requirements.

The Model

In order to build a highly available systems environment, consider the elements that contribute to availability in layers. Further, consider the incremental availability provided by each additional layer is less than the previous layer. These layers are like the tiers of a wedding cake and as such, must be built from the bottom up, largest first, in order for the cake to stand. Each successive layer contributes to the overall availability of the system and depends on the presence of all previous layers. The cost associated with this availability model is similar to a wedding cake as well. Each additional layer provides incrementally less availability, or cake, but generally costs more per layer.

The following list of availability elements is in descending order of importance and additional availability. The degree to which each feature is successfully implemented will determine the actual amount of additional availability.

1. Intrinsic System Reliability

The most important consideration in building a highly available system is to choose the system hardware that provides the highest intrinsic reliability. That is to say, choose the hardware that has the highest Mean Time Between Failures (MTBF) and the lowest Mean Time To Repair (MTTR). Both ratings must be considered. The MTBF is a reflection of the failure rate of a given system component and is used as a statement of the expected future

performance based on the past performance of a unit or population of units.

2. Operating System Resiliency

Resiliency is a measure of the operating system's ability to recover from faults. For example, HP-UX will reboot following a CPU failure and de-allocate the failed CPU during the restart process. In addition, HP-UX 11.0 will dynamically de-allocate a memory page with errors. Following an operating system crash, or panic, the system should be able to recover a memory core dump. This functionality must be enabled to allow diagnosis and subsequent repair of the problem that caused the crash. If the cause of a crash is not identified and fixed, the system remains at risk of another crash until it is.

3. Hardware Support & Repair Services

Hardware reparability is an essential component of any highly available system and is the foundation for all additional layers. Hardware components will eventually fail, regardless of the redundancy and protection mechanisms in place. The system remains at risk of another failure until a failed component is repaired. One measure of the overall availability of a system is the strength of its repair service. The response time of the support service must be considered separately from the MTTR of the various system components.

4. Redundant Disk Components

Other than printers, disks generally have the most moving parts of all components in a computer system. Traditionally, individual disk drives have higher failure rates than most other components as well. Heavily cached RAID arrays provide a disk subsystem with nearly all-redundant components, as well as hot failover and replacement, virtually eliminating downtime due to failure and repair. Such disk subsystems usually cost at least as much as all other system components combined.

5. Uninteruptable Power Supply

An uninteruptable power supply (UPS) protects what is typically one of the more vulnerable elements of a computer system, the power source. In case of a power loss, a UPS can sustain power to the system at least long enough to gracefully shutdown. It can also sustain power long enough to survive the outage. The cost of a UPS is directly related to the amount of power it must supply, and the duration that it must sustain that amount of power. The UPS can also protect the delicate circuitry of the computer system from transient

power spikes and dips.

6. Redundant Disk Links

Redundant disk links provide a dual path between the server and the disk subsystem. This feature provides protection against disk interface failure. In the event that the communications path between a server and the disk subsystem fails, the IO traffic will dynamically switch to the redundant path.

7. Network Redundancy

The network is often such an integral part of the overall application solution that when the network is down, the application is said to be down. Since a network is typically shared by more systems than just the servers running a mission-critical application it is subject to congestion and other problems beyond the control of the servers' administrators. Redundant network components reduce the risk of network failure when they are configured to dynamically failover and load balance. Usually, this kind of configuration requires specialized hardware or software to enable such functionality.

8. Software Disk Mirroring

Software disk mirroring provides redundant copies of information under the system administrator's control. This scheme provides functionality beyond that of hardware mirroring, or RAID 1, in that both copies of data are accessible to the system administrator. Conceivably, the second copy could be used for backup and recovery purposes with little or no interruption in service.

9. Journaled File System

The journaled file system provides several high availability features. The two most important features are on-line configuration changes and fast recovery from failures. The ability to grow a file system on-line is particularly important in a very dynamic environment where the need for change is difficult to predict.

10. System Failover

System failover is a software-enabled feature that allows a specified application or workload to migrate from one server to another in case of a failure. Such an event could be a system crash or an application failure. This feature is most effective during a system crash or system hardware failure since an application failure is likely to migrate with the application. System failover contributes only incrementally to the overall system availability since it protects a component of the environment that is, or should be, very reliable.

11. Policies and Procedures	Once a highly available system has been implemented, there must be a written policy that dictates the management of such a system. The policy must reflect the business needs of the application supported by the highly available system. Procedures must be written to implement this policy. Although this item appears nearly last in the list does not diminish its relative importance. Policies and procedures must be clearly defined for any level of availability, much like icing appears on every layer of a wedding cake. There must be a means to distinguish success from failure. A clearly written policy provides that distinction. Another instrument for this purpose is a Service Level Agreement (SLA).
12. Trained Support Personnel	Finally, the procedures described above must be implemented and maintained by trained system administrators. There must be at least two such individuals otherwise the administrator becomes a single point of failure. These two are like the figurine at the top of the wedding cake, without which, the cake is just another cake.

Constructing a Customized Availability Model

There are two steps in constructing a customized availability model. The first step is to establish an availability goal. This involves answering some hard questions. The answers to these questions form the basis for deciding which layers to include in a customized availability model. These are business-oriented questions, as opposed to technical questions, and should be addressed by the business managers. The following paragraphs list the most important of such questions and the ramifications of the answers.

- What are the business drivers? It is important to understand the nature of the business behind the application that runs on the highly available system. It gives a sense of reality to what otherwise could be a very cold, calculated exercise. It also helps keep the systems designer focused on the problem at hand instead of being sidetracked by trivialities.
- What is the financial impact of an outage? What does an outage cost? The cost of an outage justifies the number of layers of availability that are built into a system.
- How long can the system be down at any given time? The answer to this question is usually: “It depends”. However, the question is aimed at the busiest time, for example, during month-end close.
- How long can the system be down in a year? Since many of the availability metrics are normalized over a 1-year time frame, accumulated annual downtime is often a measure of interest.

- What, exactly is an outage? This question is more difficult to answer than it appears. The answer to this question gets to the heart of any availability measurement scheme. The next two questions help clarify the issue.
  - ✓ Is the system down when the help desk says so? If the answer here is "Yes", then the measurement of availability is localized to one spot: the help desk. However, the implication of such a measurement is that the availability calculation includes all system elements between the users and the application, including the electrical power in the building housing the users. This may not be the best indication of system availability.
  - ✓ Is the system down when the system itself reports it so? If the answer here is "Yes", then more automated mechanisms can be used to capture and calculate system availability. Further, the designer then must decide exactly which system elements will be monitored for availability and so included in the calculations.
- Should the system design revolve around the accumulated annual outage or the worst case single outage? The answer to this question lends some focus to this exercise. This choice is made based on the nature and criticality of the application running on the highly available system. Further, a system designed to handle the worst case single outage will have a substantially higher price tag.
- How high is highly available? This is the most important question to answer in building a customized availability model. The answer to this question sets a goal for the systems designer. If there is no stated goal, then there is no way of knowing if any system is highly available. All the above questions and answers lead up to this one. This question must be answered before any high availability design can be started. The answer must be a percentage, usually greater than 98%.

The second step in constructing a customized availability model is to sequence and select the availability layers that will achieve the stated goal within budget and other business constraints. First, eliminate any layers that are infeasible or unnecessary. For example, a UPS may not be needed if the data center has sufficient UPS power for the new system. Next, add any site-specific or company required layers. For example, a government agency or contractor may require full redundancy of special security devices. While the rationale behind the sequencing of the layers in the model is sound, other factors may influence the sequence of a customized model. Such factors can include, but are not limited to, company policy, already purchased equipment or software and designer's choice. The final model should have availability layers that protect against failures whose recovery time exceeds the maximum allowable downtime, as derived from the availability goal.

The following paragraphs list some questions that will help guide the systems designer through this selection and sequencing process and the ramifications of the answers.

- Which failure modes do the high availability features protect? This is the second most important question to answer in building a customized availability model. The systems designer must have a design objective in mind before building any system. This list of failure modes will become the design objective.
- Which failure modes do the high availability features not protect? The answer to this question will help refine the answer given for the previous question. It will ensure that all failure mode issues have been considered. The next two questions will help clarify this issue.

- ✓ Which failure modes do the recovery strategy and backup plan protect? Such failures can be segregated from the high availability contingencies and addressed as data recovery issues.
- ✓ Which failure modes do the disaster recovery plan protect? Such failures can be segregated from the high availability contingencies and addressed as disaster recovery issues.

- What is the total amount of time to be considered? Availability calculations are based on total usable time, that is: 365 X 24 hours minus company shutdown time in hours. For example, if a company totally shuts down during the Christmas holiday, these hours should not be included in the availability calculation, since there is no expectation of system availability during that time. This number is useful when calculating the allowable downtime in hours based on the availability goal. The allowable downtime is used to select the availability layers that will meet that goal.
- What is the total planned downtime? The planned downtime, based on the maintenance schedule, is used when calculating the allowable downtime in hours based on the availability goal. The allowable downtime is used to select the availability layers that will meet that goal.

Conclusion

Building a highly available system requires due consideration for business needs and willingness to assume risk of failure. Once business needs have been identified and the risks assessed, the systems designer should use a rigorous systematic method to build a highly available system that meets the stated needs. A layered availability model can be used to build such systems. Once an availability goal and design objective has been determined, a customized availability model can be built. Using such a customized model, the system designer can build a highly available system that meets the stated business requirements.