neverfail®
WWW.NEVERFAILGROUP.COM
PREDICT · PROTECT · PERFORM

# Eliminating End User and Application Downtime

## Continuous Availability for your Business Applications

March 2010

# Table of Contents

## Introduction

The reality in today's businesses, small and large, is that downtime of vital business applications can have a severe impact on business operations. Inevitably, any downtime for such applications leads to loss of productivity, loss of revenue, customer dissatisfaction and ultimately damage to reputation. Therefore, it should come as no surprise then that eliminating end user and application downtime, through the use of availability and disaster recovery technologies, comes in at the very top of many IT agendas.

## Server Availability vs. Ecosystem Availability

When most IT people think about availability and disaster recovery, they often think in terms of *server* availability. They typically ask themselves questions such as "How long has my Microsoft Exchange server been up and running?" and "Can my customers reach our corporate website?" The answers to these questions help IT measure against various Service Level Agreement (SLA) commitments.

But are these the right questions that we should be asking ourselves? In reality, servers seldom operate in total isolation from the rest of the IT environment. For example, a Microsoft Exchange server is typically one small piece of a much broader messaging ecosystem, which may encompass mobile messaging, instant messaging, and online collaboration services. A corporate website probably ties into various backend databases, middleware, and file servers to provide some added level of service for end-users. Even small SharePoint deployments are almost always spread across multiple servers.

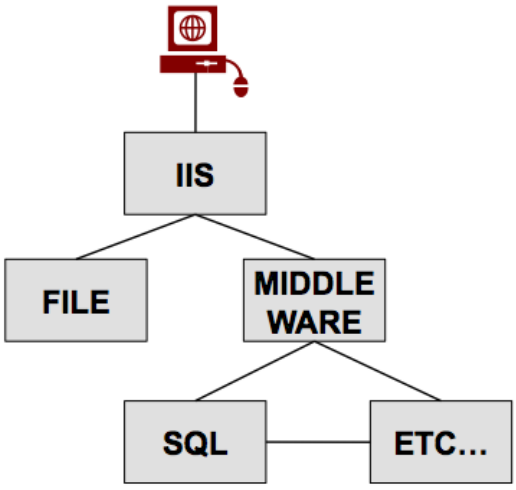


*Figure 1 – Example architecture of a multi-tier application ecosystem.*

Many home grown, "custom" business applications fall into this same category, requiring multiple, disparate components to all be healthy, in order for the end-to-end ecosystem to remain continuously available.

Whatever the business application, by taking a *server* based approach to any availability or disaster recovery project, IT administrators are adding significant complexity to their designs. This silo-based methodology almost always results in the use of several different availability / DR technologies from different vendors, with vastly different designs, capabilities and limited/no integration points. For example, an online web ordering system may use network load balancing for front-end web servers, some form of data mirroring or clustering for backend database servers, and a 3rd party availability alternative for middleware. Point-of-sale solutions, CRM tools, and even BlackBerry messaging environments follow a similar prescription, utilizing different technologies for each layer in the application stack.

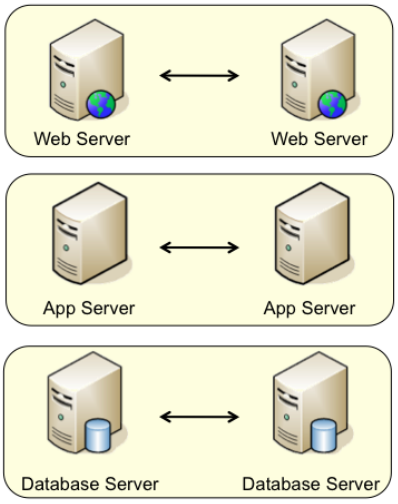


*Figure 2 – Silo-based availability methodologies protect application tiers separately.*

## Cost, Complexity, and Coordination

Taking such an approach to implementing an availability solution for your business application ecosystem has several drawbacks. First and foremost, one must consider the cost implications of utilizing differing technologies within an availability or DR architecture. The most obvious cost is the capital outlay for the hardware/software itself. By choosing (or being forced) to use different solutions from different vendors, there is no opportunity to leverage economies of scale. Most hardware and software companies offer volume-based pricing incentives for larger order sizes, but this opportunity is obviously lost when multiple alternatives from different vendors are employed. Moreover if each solution leverages different underlying hardware, disk, or OS technologies, an even greater total cost of ownership will be realized.

Of course cost extends beyond just hardware and software to include implementation, training, and ongoing management costs. Imagine deploying even a relatively simple, three-tier application architecture. In the online web ordering example discussed previously, one would need to undertake the somewhat daunting task of learning not only the intricacies of SQL clustering, but also deployment and management of network load balancing and any middleware components necessary. Each time a new version of any of these solutions is released, there's the added cost of relearning a new technology.

Next consider the complexity of integrating disparate availability technologies from multiple vendors. Are they guaranteed to interoperate with one another? Is such interoperability built-in (unlikely) or will some level of customization and manual scripting (very likely) be required, so that each tier can communicate with the other tiers? If custom scripting is required, what happens when even a single piece of the availability architecture changes? Will additional, custom consulting work be required to update and re-test existing scripts? Last but not least, if and when something breaks down, whose responsibility is it to determine the root cause? With different solutions from different vendors, one must be wary of the inevitable finger-pointing that may result when things go wrong. Of course one alternative is to simply not integrate the solutions – after all, so long as each piece is doing its job, isn't it safe to assume that the entire system is operational? Not necessarily.

Consider for example the deployment of a multi-tier, distributed architecture across physical sites for DR purposes. If the entire, primary production site fails, will the servers start up in the proper order and fashion at the remote site, or will some level of interaction be required from an administrator? Now consider the more likely type of failure – when just one component rather than an entire site fails. Unless you've deployed a combined HA + DR solution, chances are that the single failed component will resume operations at the DR site. But in most cases, the latency between sites will be too high for any multi-tier application to function properly. In this scenario, it's best to actually fail all of the components across to the remote site as a single, cohesive unit. But once again, how does this coordination take place? Either we're back to scripting the failover in some manner, or else some hands-on administrator involvement is required. When that happens, recovery times inevitably increase; when recovery times increase, so does the bottom-line cost of the outage to the business.

## Protecting the Entire Business Application Ecosystem

Considering the costs and complexities evident in an application-specific, silo-based, multi-vendor availability methodology, surely a better approach would be to source a complete solution, from a single vendor, that can protect your entire business application ecosystem.

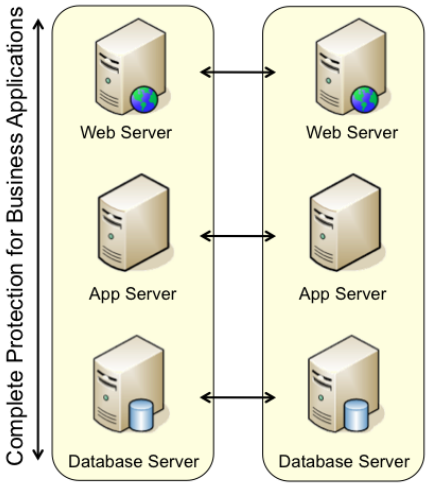


*Figure 3 – Protecting the entire business application ecosystem with a single solution.*

In doing so, capital costs are minimized as well as training and implementation costs – after all, buying and learning one solution is far cheaper and simpler than two, three, or more. Integration complexity and inter-tier failover coordination is also ideally eliminated, assuming of course that the selected technology vendor has developed this into the core of their product. That's where Neverfail comes in.

## Complete, Continuous Availability for Business Applications

Neverfail provides a complete suite of continuous availability and DR products, all fully integrated with one another, via the Neverfail Continuous Availability Director.

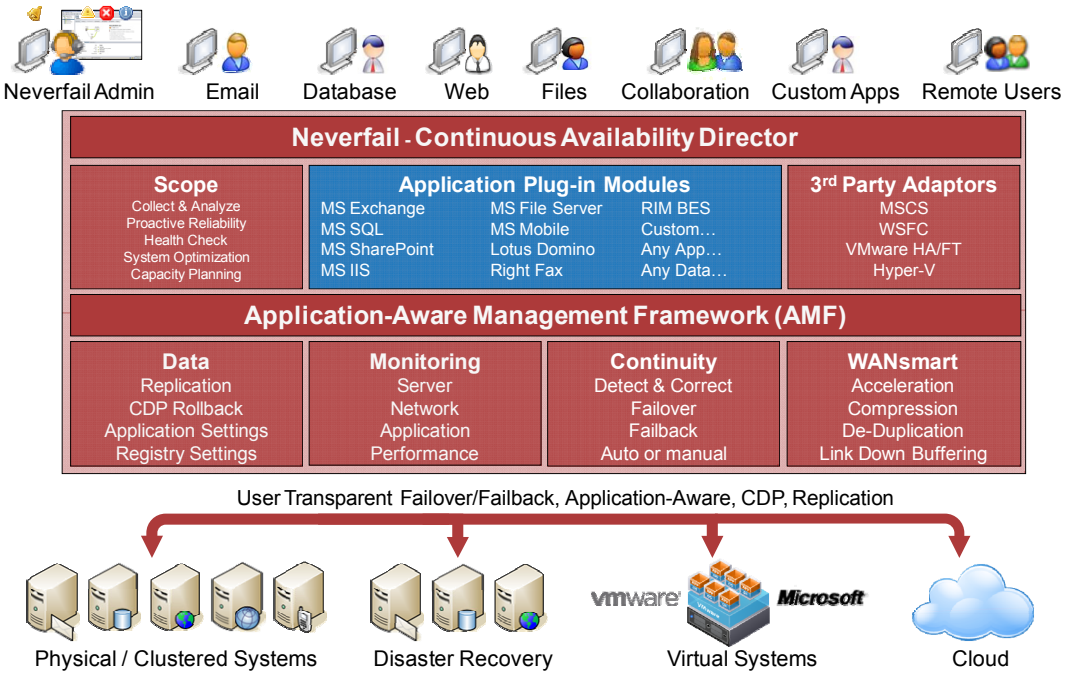


*Figure 4 – The Neverfail Continuous Availability Suite.*

To eliminate user and application downtime, the Neverfail solution has been architected from the ground up to protect data through continuous replication, to monitor the health and state of business applications and to enable automated failover, or manual switchover, to additional local and/or remote servers.

The Continuous Availability Director provides an enterprise-wide, business-centric view of critical applications and IT services. It has a flexible approach, which allows logical grouping of application, database, messaging and other servers. The grouping provides a way of visualizing interdependencies across servers and gives a central console where events, alerts and the overall health can be viewed.

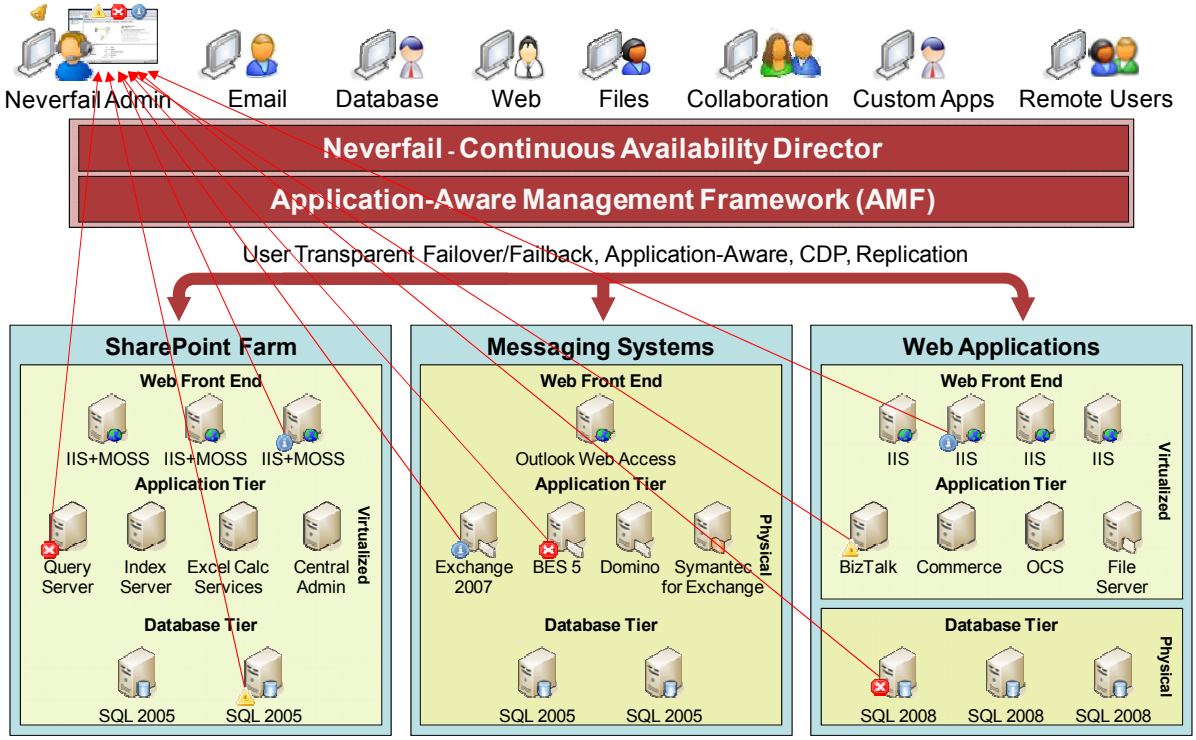


*Figure 5 – Protecting multiple applications with the Neverfail Continuous Availability Director.*

Neverfail's continuous availability solution is unique, in that it can proactively manage the availability of entire business application ecosystems. It protects critical application availability against physical server hardware, network infrastructure and operating system and application failures. If a problem occurs, Neverfail can take a variety of pre-emptive, corrective actions including fully coordinated failover of all components within the ecosystem.

The net result is the elimination of end-user and continuous availability for business applications.