

PUSHBUTTON RECOVERY: STILL A DREAM

BY JON WILLIAM TOIGO



The holy grail of disaster recovery and business continuity is an idea best described as “pushbutton recovery.” In a perfect world, we would be able to push a button to shift business operation workload and user connections to an alternative location should our primary business environment become untenable.

To some extent, this can be done within a local environment today using “failover clusters” of servers, each tethered to its own storage repository containing “mirrored” data. However, the metaphor does not currently extend into the realm of geographically-dispersed sites.

There are many technical reasons (aside from cost) why wide area failover is such a challenge. For simplicity, these hurdles can be divided into three categories: application layer issues, hosting platform issues, and storage issues.

Application Layer Issues

In most companies, there are a mix of home grown and shrink wrapped software applications supporting mission critical business processes that may or may not be instrumented for failover based recovery over distance. Sprawling client-server applications may have been rolled out over a lengthy period under the auspices of many different project managers, each with his or her own idea about application architecture. These apps may be built on several layers of server infrastructure – providing functions such as web serving, application code hosting, and database hosting – all held together by a mix of networks, fabrics and middleware. Recovery or failover of these applications requires recovery of the entire hosting environment.

A common problem in planning for the failover of such n-tier applications is linked to the way that distributed application components communicate with one another. Middleware is the glue that manages message passing between client-server components and there are many different types of middleware in the market.

One dominant type of middleware uses “hard coded” connections between application components, which is to say that messages are directed between application components using unique identifiers such as machine IDs or IP addresses assigned to the hardware where the application component resides or built into hardware devices themselves. Remote Procedure Calls (RPCs) are an example of a hard coded middleware approach.

Alternatively, message-oriented middleware (MOM) can be used to link together application components in a non-hardware dependent way. MOM typically creates a directory service that identifies the locations of application components. This service is used by applications to look up the location of a component with which it needs to communicate and to direct messages to that component.

The difference is profound in terms of its implications for application failover: hard-coded middleware requires planners to replicate every piece of physical infrastructure in the primary site on a one-for-one basis in the recovery

environment. Not only can this be prohibitively expensive, it can also be a bear to maintain over time and in the face of almost constant change.

By contrast, using MOM to interconnect application components enables applications to be re-hosted with less worry about the characteristics of their physical host locations. MOM simply discovers its application components, lists them in a directory or table, and plays “postal worker” from that point forward to ensure that messages go where they need to go.

So why haven’t application developers used MOM instead of RPCs all along? Usually, it is because no one ever asked them to or explained the advantages from a recovery standpoint of doing so. DR/BC planners are rarely part of an application design team.

Going to commercial or shrink-wrapped software doesn’t necessarily solve this situation. In many cases, commercial software packages are also cobbled together from many developers using whatever middleware connections are popular, available or least expensive at any given time.

There is no easy way to surmount the hurdle to failover recovery of apps that are not designed with failover in mind, of course. And making matters worse is the fact that there is no provision in the design of most applications for something called a “heartbeat” or checkpoint that can be monitored to determine when the application has stopped and failover is required.

Hosting Layer Issues

Below the layer of the application is the layer of the hosting environment, which includes the operating systems of servers, additional services such as clustering and virtualization, and connections to storage infrastructure where the data from applications resides. You don’t need to be an expert IT architect to know that server virtualization is a hot button this year, as failover clustering has been for the past five years or so. These are key elements of mission critical hosting infrastructure for reasons of application availability, performance, cost reduction and improved management. They do not, however, provide much support for distance failover

and may in fact impede your ability to fail applications over distance.

Clustering, in the popular usage of the term, refers to relationships established between a pair of servers to enable workload to shift to one server if the other fails. We won’t go into the nuances of difference between active-active and active-passive clusters. Suffice it to say, clustered configurations are typically implemented to provide a measure of resiliency against the failure of one member of the clustered pair in the production shop.

Server virtualization is another popular technology for application hosting intended to combine multiple physical servers into one by creating virtual machines inside a physical server. The advantages touted by virtualization vendors are many, but they do not currently extend to failover over distance.

What physical server clustering and server virtualization have in common is that they add more complexity to the disaster recovery process (not to mention adding more potential points of failure). In a growing number of cases, companies seek to failover their carefully crafted clustered hosting or virtualized server installation in toto at a remote site. When this is the goal, they too often discover that such failover requires a one-for-one replication of all of the hardware in the primary setting at the recovery site to ensure success.

Another twist on this story: sometimes the backup site provider can only offer a virtual server environment to its customers. This imposes a requirement on companies that are using physical servers or clusters in their production shops to redesign their application hosting environment “on the fly” if they want to failover their mission critical hosts. The only option is to find another hot site vendor.

Why isn’t host architecture being deployed with attention to its suitability for wide area failover? Too often, DR/BC planners have no input to the infrastructure design or acquisition process.

Storage Layer Issues

Storage remains the one industry segment that has not succumbed to commoditization. Arguably, servers

CONTINUED ON PAGE 14



and network equipment are commodity products today and one product can be readily substituted for another, regardless of manufacturer. In the case of storage equipment, vendors have followed a path to market first publicly articulated by former EMC CEO Ruettgers in 2001: if all vendors are selling a box of Seagate hard disks, the only way to differentiate platforms and to sustain profit margins is to join value add software at the hip to proprietary controllers in array products that lock in the consumer and lock out the competition.

This game plan provides significant impediments to cost-effective failover over distance. For each vendor's wares, a data mirroring process exclusive to the vendor, must be implemented – and only between identical storage products from that vendor. In a mixed or heterogeneous storage infrastructure, this process of “copy after write” can get extraordinarily expensive. Questions are also being raised regarding the visibility of the mirroring process: can you be certain that the right data is being mirrored and that it is being mirrored correctly – without disrupting the mirroring process to find out? The answer is no.

Another challenge is the complexity of the storage environment. So-called storage area networks (SANs) have provided a means to scale storage capacity dramatically in order to host more electronic data. However, it has also led to the creation of huge “junk drawers” within many companies that make it difficult or impossible to identify what data belongs to which application.

Finally, SAN storage is a “standards free” world, or rather one characterized by standards that do not assure product interoperability. For example, two SAN switch vendors can implement switch designs that embody the letter of the ANSI standards for switches with absolute certainty that their switches will not talk to one another.

These three issues, just a subset of a broader problem set, create huge impediments to pushbutton recovery both from a strategy efficacy and strategy cost perspective. A simpler approach for data mirroring would be a “copy on write” strategy in which data from applications is written to both a local and a remote storage target concurrently. This can be done at the storage protocol level using UDP/IP based technologies like those proffered by Zetere Corporation in Irvine, CA. It can also be accomplished using some of the better storage virtualization products, such as SAN Melody and SAN Symphony from DataCore Software. Copy on write can also be done by a switch or a specialty host bus adapter.

Why isn't this being done? To be honest, most DR/BC planners are not particularly storage technology savvy, and even those who are seldom receive invitations to storage planning or acquisition meetings.

What Can Be Done

Pushbutton recovery, if it is to be realized, will require two things. The first is a greater measure of involvement by DR/BC professionals in the day to day decisions of IT, including application

design, product acquisition and data replication strategy. The DR person needs to get a seat at the table, which in turn requires that he or she become more conversant in the technologies under consideration. This may be a challenge considering the widespread view that DR planners don't need to be rocket scientists to do their jobs, just practical project managers. The reality is very different.

The second requirement for pushbutton recovery is the selection and deployment of failover tools that I call “wrappers.” Wrappers are software products that encapsulate all of the infrastructure associated with an application – or better yet, a business process – and provide the means to build scenarios for its failover that can be tested without interfering with production work and executed either automatically or with minimal human intervention if the need arises.

Some wrapper products are CA XOsoft and Neverfail Group's Neverfail. They enable the creation of failover scenarios in the face of complex infrastructure and provide the means to coordinate many disparate recovery processes, including storage level mirroring and application layer transaction recording. Another promising product is Continuity Software's RecoverGuard, which provides a dashboard that continuously monitors the capability of recovery strategies to meet recovery time and recovery point objectives as changes occur in the infrastructure or the volume of data that needs to be replicated.

With so much working against “pushbutton recovery,” these products hold out hope that our always on business processes will eventually be supported by always on protection services.

ABOUT THE AUTHOR

Jon William Toigo is CEO and Managing Principal of Toigo Partners International LLC, an independent consultancy and technical research & analysis firm focused on providing actionable guidance to IT decision makers. He is a 25+ year IT veteran who has worked both as an operative within corporate information systems departments and as a senior consultant with two international systems integrators. He can be reached at www.toigopartners.com or jtoigo@toigopartners.com