THE EXPERT'S VOICE® IN SHAREPOINT

SECOND EDITION

Pro SharePoint Disaster Recovery and High Availability

BE PREPARED FOR WHEN BAD THINGS
HAPPEN TO GOOD PORTALS

Stephen Cummins

Apress®

For your convenience Apress has placed some of the front matter material after the index. Please use the Bookmarks and Contents at a Glance links to access them.

friendsof ED™

Apress®

Contents at a Glance

Introduction

I wrote this book to share what I have learned about high availability and disaster recovery for SharePoint at this point in time. It is certainly an interesting time. In the past 10 years, SharePoint has gone from a compiled application that just looked superficially like a web application into a more fully fledged cloud platform. The process is far from over, however, and SharePoint will likely look very different in 10 years time. But there is no doubt in my mind that it will still be in use in some form. It will be interesting for me look back on this book and see what's the same and what's different. I tried to focus on general principles in this book so that even as the technology changes, the principles still apply.

The main risk with any information recording system is that once you use it, you become dependent on it. If that information becomes unavailable for any number of reasons, it has a detrimental effect on your organization. We are just as subject to whims of Mother Nature as we ever were, and now technology has become complex enough that it is difficult for anyone but the most specialized to know enough about it to know how to make it resilient, redundant, and recoverable. In relation to SharePoint, this book will give you the knowledge and guidance to mitigate this risk.

Who This Book Is For

If you worry about what would happen to your organization if the data in your SharePoint farm was lost, this book is for you! It is a technical book in parts, but most of it is about the principles of good planning and stories of how things have gone right and wrong in the field. My intention is that it should be instructive and entertaining for anyone whose organization has begun to rely on SharePoint to function.

How This Book Is Structured

Each chapter describes practical steps that can be taken to make your system more resilient and give you the best range of options when a disaster hits your SharePoint farm. Reading, however, is not enough. I offer pointers to inspire you to take what you have learned here and apply it in the real world. After you read each chapter, put into practice what you have learned! At the very least, take notes of your thoughts on what to do so you can do it later.

Chapter 1: Steering Away from Disaster

To protect your content, you must know your technology and realize its importance to your organization. Roles must be assigned and responsibility taken. Moreover, there should be a way to record near-misses so they can be captured and addressed. SharePoint is not just a technology platform; it's partly owned by the users, too. They and management must play a part in its governance.

Chapter 2: Planning Your Plan

Before you can write a plan you will need to lay a foundation. You will first need stakeholder and management buy-in. You will also need to do a business impact assessment. You may need to plan different SharePoint architectures that have different RTO/RPOs and different cost levels relative to the importance of the data within them. You will also need to create a good SLA and plan how to coordinate a disaster.

Chapter 3: Activating Your Plan

Many processes and procedures have to be in place before you can put your SharePoint disaster recovery plan into action. These are not abstract things on paper; they are actual tasks that defined roles have to perform. This chapter details who is going to do what and when, knowing the interdependencies, accessing the plan, and making sure in advance the plan contains what it should.

Chapter 4: High Availability

High availability is something achieved not just through meeting a percentage of uptime in a year. It is a proactive process of monitoring and change management to ensure the system does not go down. It is also about having high quality hardware. Finally, it is about having redundancy at every level of your architecture from the data center down to the components of the individual service applications.

Chapter 5: Quality of Service

The main ways to improve your quality of service are WAN optimization, designing your farm so that content is near the people who need to see it, and caching infrequently changed pages. WAN acceleration can only help so far with the limitations of latency, but there are options in SharePoint 2013 to get a cost-effective compromise between user satisfaction and a not overly complex architecture.

Chapter 6: Back Up a Step

Your farm is a unique and constantly changing complex system. When focusing on how to back up and restore it successfully, you will need clearly documented and tested steps. You can't fully rely on automated tools, partly because they can't capture everything and partly because they can only capture what you tell them to and when.

Chapter 7: Monitoring

SharePoint must be monitored at the Windows and application levels. The SharePoint application is so dependent on the network infrastructure that anything wrong with SQL Server, Windows, or the network will affect SharePoint. The information in this chapter gives you the guidance and direction you need to watch what needs watching.

Chapter 8: DIY DR

This chapter shows that the task of maintaining backups of valuable content need not be the exclusive domain of the IT staff. Giving users the responsibility for and means to back up their own content is an excellent idea from an organizational point of view as it is likely to save resources in both backup space and IT man-hours.

Chapter 9: Change Management and DR

Change management is a collaborative process where the impact of change has to be assessed from a business and a technical perspective. Change is the life-blood of SharePoint; without it the system succumbs to entropy, becomes less and less relevant to user needs, and becomes a burden rather than a boon to the business.

Chapter 10: DR and the Cloud

Analyze the additional problems and opportunities presented by off-premises hosting. There is still a great deal of planning involved in moving to the cloud. This chapter looks at the process by which SharePoint developed into its current form, how cloud architecture options come down to cost and control, and how multi-tenancy and planning federation are key aspects of SharePoint in the cloud.

Chapter 11: Best Practices and Worst Practices

When it comes to best and worst practices in SharePoint, there is no such thing as perfection and no implementation is all bad. But it is possible to improve and to avoid obvious pitfalls. Primarily, you have to avoid the easy path of short term results, the quagmires of weak assumptions, a reactionary approach to change, and an irresponsible approach to governance. Those four principles will get your SharePoint platform off to a good start and keep it on course.

Chapter 12: Final Conclusions

This chapter brings together the key principles contained in this book. The approach has been to create a guide that can be used in any circumstance rather than to define only one approach. Principles are more universal and can be applied to any version of SharePoint irrespective of changes in the underlying technology. Even as SharePoint transitions to the cloud, there are still lessons than can be applied from the four previous versions of SharePoint, and high availability and disaster recovery in general.

CHAPTER 1

Steering Away from Disaster

On my very first SharePoint job back in 2001, I spent hours backing up, copying, and restoring the SharePoint installation from an internal domain to the one accessible to users from the Internet. This was not a backup strategy; it was a crude way to get content to the Internet while keeping the intranet secure. But it made the system very vulnerable to failure. Every time content was updated, I had to manually overwrite the production SPS 2001 with the updated staging SPS 2001 out of hours so users could see the changes the next day. This started to become a nightly occurrence. I still remember the feeling of fear every time I had to run the commands to overwrite the production farm and bring it up to date. I would stare at that cursor while it made up its mind (far too casually, I thought) to bring everything in line. I would sigh with relief when it worked and I was able to see the changes there. I still feel the sense of mild panic when it didn't work and I had to troubleshoot what went wrong. It was usually an easy fix—some step I missed—but sometimes it was a change to the network or the Exchange server where the data was stored or a Windows security issue.

Disaster was always only a click away, and even back then I knew this way was not the best way to do what I was doing. It made no sense, but I did it every day anyway. The process had been signed off by management, who thought it looked secure and prudent on paper, but in reality it was inefficient and a disaster waiting to happen. Eventually, I left for a better job. Perhaps that's how they still do content deployment there.

Maybe you are in a similar situation now: you know that the processes and procedures your organization is using to protect itself are just not realistic or sustainable. They may, in fact, be about to cause the very thing they are supposed to protect against. Or perhaps the disaster has already occurred and you are now analyzing how to do things better. Either way, this book is designed to focus your thinking on what needs to be done to make your SharePoint farm as resistant to failure as possible and to help you plan what to do in the event of a failure to minimize the cost and even win praise for how well you recovered. The ideal scenario is when a disaster becomes an opportunity to succeed rather than just a domino effect of successive failures. Can you harness the dragon rather than be destroyed by it?

This chapter addresses the following topics:

- The hidden costs of IT disasters.
- Why they happen.
- Key disaster recovery concepts: recovery time objective and recovery point objective.
- Key platform concepts: networks, the cloud, IaaS, and SaaS.
- Roles and responsibilities.
- Measures of success.
- Some applied scenarios, options, and potential solutions.

The Real Cost of Failure

This book focuses on two different but related concepts: high availability (HA) and disaster recovery (DR). Together they are sometimes referred to as Service Continuity Management (SCM). While SCM focuses on the recovery of primarily IT services after a disaster, as IT systems become more crucial to the functioning of the business as a whole, many businesses also assess the impact of the system failing on the organization itself.

No matter what your core business, it is dependent on technology in some form. It may be mechanical machinery or IT systems. IT systems have become central to many kinds of businesses, but business managers and owners have not kept up with the pace of change. Here's an example of how core technology has become important for many types of companies.

In 2008 Starbucks closed all its US stores for three hours to retrain baristas in making espresso. It cost them $65 million in lost revenue. Was that crazy? They did it on purpose; they realized the company was sacrificing quality in the name of (store) quantity. They had expanded so fast that they were losing what made the Starbucks brand famous: nice coffee in a nice coffee shop. They anticipated their seeming success in the short term would kill them in the long term. They had more stores, but fewer people were coming in. The short-term cost of closing for three hours was far less than what they would lose if they did not improve a core process in their business. Making espresso seems a small task, but it's one performed often by their most numerous staff members. If those people couldn't make a quality espresso every time, the company was doomed in the longer term. Focusing on this one process first was a step in improving business practices overall. It was a sign that Starbucks knew they need to improve, not just proliferate, in order to survive.

In this case, falling standards of skill was seen as a reason to stop production. It was planned but it underlines the cost when a business can't deliver what they produce. Your SharePoint farm produces productivity. It does this by making the user activity of sharing information more efficient. SharePoint is worthless if the information in it is lost or the sharing process is stopped. Worse than that, it could seriously damage your business's ability to function.

Perception is reality, they say. Even if only a little data or a small amount of productive time is lost, some of an organization's credibility can be lost as well. A reputation takes years to build but it can be lost in days. If increasingly valuable information of yours or your customers is lost or stolen from your SharePoint infrastructure, the cost can be very high indeed. Your reputation might never recover.

Poor perception leads to brand erosion. IT systems are now an essential part of many businesses' brand, not just hidden in a back room somewhere. For many companies, that brand depends on consumer confidence in their technology. Erosion can mean lost revenues or even legal exposure. The 2011 attack on Sony's PlayStation Network where 100 million accounts were hacked (the fourth biggest in history) will cost Sony a lot of real money. One Canadian class action suit on behalf of 1 million users was for $1 billion. What might the perceived antenna problems with iOS4 have cost Apple if they had not reacted swiftly (after some initial denial) to compensate customers?

Large companies like Starbucks, Sony, and Apple know technology is not just part of what they sell, it is core to who they are. If you neglect the core of your business, it will fail. The cost of total failure is much higher than the cost of understanding and investing in the technology that your staff relies on every day. SharePoint has become more than a useful place to put documents in order to share them with other users. It is now the repository for the daily tasks of many users. It has become the core technology platform in many businesses and it should be treated as such.

Why Disasters Happen and How to Prevent Them

In IT there is a belief that more documentation, processes, and procedures means better documentation, processes, and procedures—like the idea that more Starbucks meant Starbucks was doing better. In fact, the opposite is true. Processes around HA and DR (indeed all governance) should follow the principle that perfection is reached not when there is nothing left to add, but when there is nothing left to take away. Good practice requires constant revision and adjustment. Finally, the people who do the work should own the processes and maintain them. In too many businesses the people who define the policies and procedures are remote from the work being done, and so the documents are unrealistic and prone to being ignored or causing failures.

Success/Failure

SharePoint farms are like any complex system: we can't afford to rely on the hope that haphazard actions will somehow reward us with a stable, secure collaboration platform. But the reality is most of our processes and procedures are reactive, temporary stop-gap solutions that end up being perpetuated because there's no time or resources to come up with something better. We would, in fact, be better off with "Intelligent Design" than with Evolution in this case because we are in a position to interpret small events in a way that lets us anticipate the future further ahead than nature. At the same time, near misses dangerously teach us something similar but opposite: if you keep succeeding, it will cause you to fail. So who is right and how can we apply this to the governance of our SharePoint architectures?

There is some research from Gartner that has been around for a few years that says that we put too much emphasis on making our platforms highly available only through hardware and software, when 80% of system failures are caused by human error or lack of proper change management procedures. So, what are the thought processes that lead us to ignore near misses and think that the more success we have, the less likely we are to fail?

If we're not careful, success can lead to failure. We think that because we were lucky not to fail before, we will always be lucky. Our guard goes down and we ignore the telltale signs that things will eventually go wrong in a big way, given enough time.

Research shows that for every 30 near misses, there will be a minor accident, and for every 30 of those, one will be serious. SharePoint farms have monitoring software capturing logs, but they only capture what we tell them to; we have to read and interpret them. The problem is that not enough time is allocated to looking for small cracks in the system or looking into the causes of the near misses.

But a more pernicious cause of failure is the fact that when processes are weak, the people who monitor the system are continuously bailing out the poor processes. Those who have responsibility for the processes are not reviewing the processes continually to keep them up to date. The people who don't own the process are not escalating the problems; instead they are coming up with quick fixes to keep things going in the short term. Sooner or later, they will get tired or frustrated or bored, or they'll leave before things really go wrong. Then it will be too late to prevent the real big FUBAR.

Thus, management must not ignore the fact that staff on the ground are working at capacity and keeping things going, but it will not last. Likewise, staff on the ground must step up and report situations that will lead to system failure and data loss.

Is failure necessary for success? I think that every process has to be the best it can be with the realization that it must be tested and improved continuously. This is the essence of governance: people taking ownership of change and reacting to it constructively. The constant evolution of policies is needed.

Your SharePoint Project: Will It Sink or Float?

Let's use an analogy—and it's one I will revisit throughout this book. Your SharePoint project is like the voyage of a cruise liner. Will it be that of a safe, modern vessel or the ill-fated Titanic? Your cruise ship company has invested a lot of money into building a big chunk of metal that can cross the Atlantic. Your SharePoint farm is like that ship. The farm can be on premises, in the cloud, or a hybrid of both. You have a destination and high ambitions as to what it will achieve. You know for it to succeed you will need an able crew to administer it plus many happy paying passengers.

This analogy is assuming something inevitable. The ship will sink. Is it fair to say your SharePoint implementation will fail? Of course not, but you should still plan realistically that it could happen. Not being able to conceive of failure is bound to make you more vulnerable than if you had looked at everything that could go wrong and what should be done if it happened. This is why ships have lifeboat drills—because they help prevent disaster. Acknowledging the fact that disasters do happen is not inviting them. In fact, it does the opposite; it makes them less likely to happen as it helps reveal weaknesses in the infrastructure and leads to realistic plans to recover more quickly when disasters do happen.

Figure 1-1 is of a typical SharePoint farm. Note that more than half of the servers are redundant. The farm could still function if one web front end, one application server, and one SQL server stayed functioning. Let's return to the Titanic metaphor. It was engineered with a hull with multiple compartments; the builders said that the ship could still float if many of these were breached. In fact, ships had hit icebergs head on and survived because of this forethought in the design.

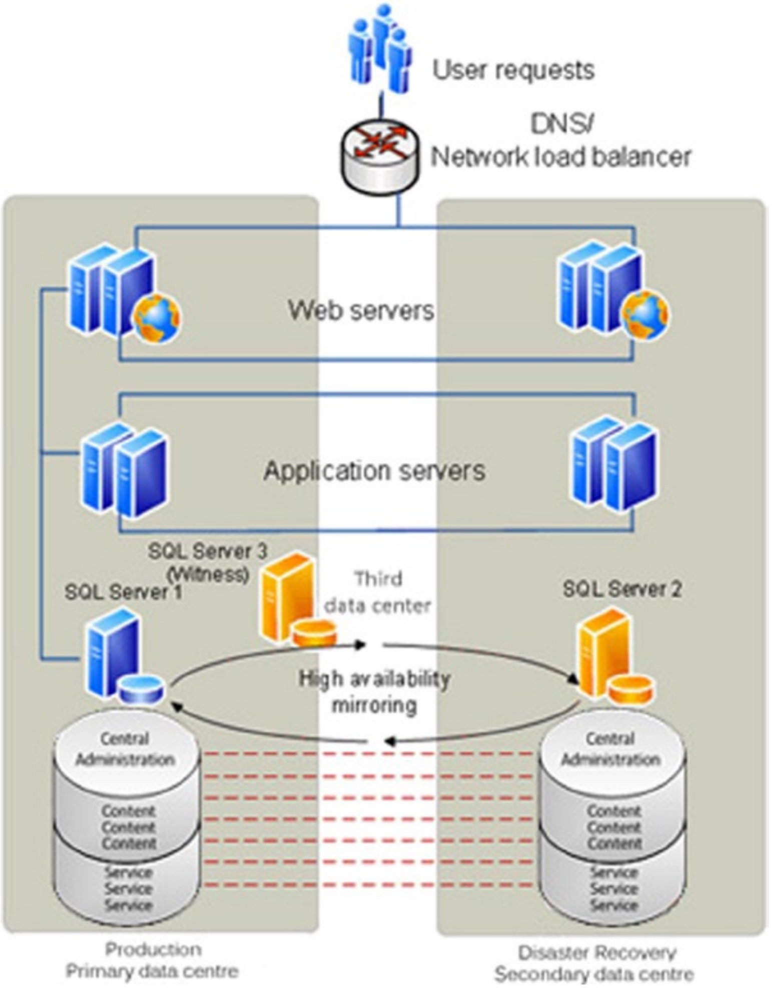


Figure 1-1. *Typical highly available on-premises SharePoint farm*

So technology convinced experts that very large ships were beyond the laws of physics. It somehow became widely believed that not only was this the biggest, most luxurious liner on the sea, but it was also virtually unsinkable. And we all know how that turned out. The story was very sensational news at the time and still is. The press today is no different from the press 100 years ago; they love big stories. The Titanic was such a compelling story because it was the world's biggest passenger ship on its maiden voyage full of the rich and the poor—a metaphor for modernity and society.

Perhaps your SharePoint deployment will be watched by the press, too, and you will want it to go well for the same reasons. Perhaps it will only be watched by internal audiences, but its success or failure will still be very visible as it involves all kinds of users in your organization. This is certainly a good argument for piloting and prototyping, but the real full-scale system still has to go live and set sail someday.

High Availability: The Watertight Compartments

High availability is the IT terminology for the efforts made to ensure your SharePoint Farm will not sink, no matter what happens to it—its resilience and quality can handle the damage and still keep afloat. Automatic systems that kick in when things go wrong are referred to as *failover systems*. In the case of my analogy, they would be like the bulkhead doors that close to make the compartments watertight (see Figure 1-2). These could be triggered manually but would also kick in automatically if water rose to a certain level in the compartments. In SharePoint, clustering, load balancing, and mirroring provide this failover and resilience. But they can be overwhelmed.

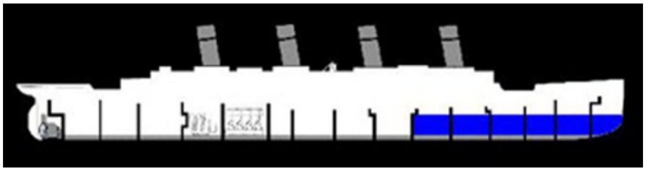


Figure 1-2. *High availability on R.M.S. Titanic*

In most IT systems, it's too easy to provide the minimum or even recommended level of resilience without much active thought. In the Titanic, the 16 compartments exceeded the Board of Trade's requirements; the problem was that 16 watertight cubes in a ship are inconvenient for the crew (administrators) and passengers (business users). There were many doors between the compartments so that people could move freely through these barriers. As a result, safety was trumped by convenience. This is a common reason for the failure of high availability systems in SharePoint, too. The failure is usually in the rush to apply updates and routine improvements to the system. The more complex the high availability systems, the more moving parts there are that can fail.

In a SharePoint farm, you can achieve high availability through a number of options. A combination of the following is common. Each will be explained in more detail later in the book:

- *SQL mirroring:* Synchronously maintaining a copy of your databases. *Synchronously* means the data is always the same at the same time.
- *SQL clustering:* Spreading a SQL instance over multiple machines. An *instance* is a group of servers that appears as one SQL server.
- *SQL log shipping:* Backing up to file the data and restoring to another SQL instance asynchronously. *Asynchronously* means the data is not exactly the same at the same time. There is a delay of hours in moving the logs from one instance to the other.
- *Multiple data centers (DCs):* This means locating your server farms in independent premises in different geographical locations. For example, Office 365 for EMEA is in Dublin, but there is also another DC in Amsterdam.
- *Load balancing:* Software or hardware, more than one server seems to have the same IP address as they have virtual IP addresses.
- *Stretched farm:* Hosting some servers in your farm in different data centers.

- *SAN replication:* Synchronously maintaining a copy of your data.
- *Redundant disaster recovery farm:* A second farm in another location ready to take the place of the production farm. This will be explained in more detail later in the chapter.
- *Availability zones and regions:* Used in Amazon Web Services, these are analogous to servers and data centers.

Disaster Recovery

Disaster recovery is what to do when something has already gone wrong. With a SharePoint farm, it's the point when users start to lose access, performance, or data. It can also be when security is compromised. Basically, it's when the integrity of the system is compromised. You've hit the iceberg. With the Titanic, the disaster recovery process was the lifeboat drill and the lifeboats themselves. With a SharePoint farm, it's the processes, policies, and procedures related to preparing for and undergoing a recovery from a disaster. Thus, it is the planning that goes into what to do from the point the problem is detected. Note that it may not be the exact time the problem started to occur—only when it is detected. Error detection and reporting will be examined in further detail in chapter 7.

On the Titanic there were not enough lifeboats because it was believed that the ship was unsinkable due to its watertight compartments. Also, it was believed that it would take the crew too long to load all the lifeboats in the event it was sinking (the Titanic had a capacity of over 3,500 souls, although there were only about 2,500 on board when it sunk). Finally, the regulations were out of date at the time; the ship was legally compliant, but in actuality had less than half the capacity needed, even if the lifeboats had been full. Relying too much on documentation and the recommended approach is not always enough.

Recovery Time Objective and Recovery Point Objective

Two metrics commonly used in SCM to evaluate disaster recovery solutions are recovery time objective (RTO), which measures the time between a system disaster and the time when the system is again operational, and recovery point objective (RPO), which measures the time between the latest backup and the system disaster, representing the nearest historical point in time to which a system can recover. These will be set in the Service Level Agreement (SLA), which is the legal document the provider has to follow. For example, SharePoint Online as part of Office 365 has set an RPO and RTO in the event of a disaster as the following:

> *"12-hour RPO: Microsoft protects an organization's SharePoint Online data and has a copy of that data that is equal to or less than 12 hours old.*
>
> *24-hour RTO: Organizations will be able to resume service within 24 hours after service disruption if a disaster incapacitates the primary data center."*

However this is only the Standard level, Enterprise level is higher:

> *"1 hour RPO: Microsoft protects your SharePoint Online data and has a copy of that data that is equal to or less than 1 hour old.*
>
> *6 hour RTO: Organizations will be able to resume service within 6 hours after service disruption if a disaster incapacitates a hosting data center."*

Networks and the Cloud

Think of your network or the cloud as the ocean. It's big, unpredictable, and full of dangerous things, most of which the administrator can't control. There are denial-of-service attacks, human error, hardware failures, acts of God, and all manner of things that can happen to compromise your system. Later in Chapter 4 I will describe the kinds of events that can compromise the integrity of your system and how to mitigate them.

IaaS vs. SaaS

Infrastructure as a Service (IaaS) and Software as a Service (SaaS) emphasize high availability over disaster recovery. Naturally, it makes more sense to keep the system working rather than recover from it failing. With IaaS, high availability is more in the hands of the tenant. With SaaS, like SharePoint Online in Office 365, you are more reliant on the provider to keep the system working. My analogy is that IaaS is like being a crew member; you have training and responsibility to keep the passengers safe. With SaaS, you are more like a passenger, reliant on the provider to keep you safe.

For example, in the case of an IaaS provider like Amazon Web Services (AWS), there is the ability of the tenant to place instances in multiple locations. These locations are composed of regions and availability zones. Availability zones are distinct locations that are engineered to be insulated from failures in other availability zones and provide inexpensive, low latency network connectivity to other availability zones in the same region. Think of these as your watertight compartments.

By launching instances (in your case, your SharePoint servers) in separate availability zones, you can protect your applications from the failure of one single location. There are also regions. These consist of one or more availability zones, are geographically dispersed, and are in separate geographic areas or countries. By spreading your instances across these, you have greater resilience.

With SaaS examples like Office 365, if there is a problem with the platform, you have less control over reacting to that problem. Think of this as a passenger bringing his or her own lifejacket. I will go into more detail in Chapter 10 on how to have more control.

SharePoint in the Cloud

The IT world is shifting to where computing, networking, and storage resources are migrating onto the Internet from local networks. SharePoint is a good candidate for cloud computing because it is already web-based. From a setup and administration point of view, it has a growing, complex service architecture. Also, many companies would gladly do without the cost of having the skills in house to administer it, not to mention the opportunity to move Exchange to the cloud. This will not happen all at once, but it does mean that hosting your SharePoint farms on premises is no longer the only option. For that reason I will outline the new cloud options for those unfamiliar with them.

Once upon a time a picture of a cloud was used on network diagrams to denote the Internet (see Figure 1-3). This is why we use the term *the cloud* now. It had a "Here be dragons" feel about it. (Prior to Europeans discovering big chunks of the world, large areas on maps were labeled "Here be dragons," as shown in Figure 1-4. It was a way to fill an empty space that could not be understood. With this lack of knowledge comes fear; hence pictures of dragons.) In the context of this metaphor, the dragon is complacency—a false bravado born of fear. The cloud is full of positive benefits for businesses. It will soon be seen as a New World to be discovered and explored, not an unknown danger.

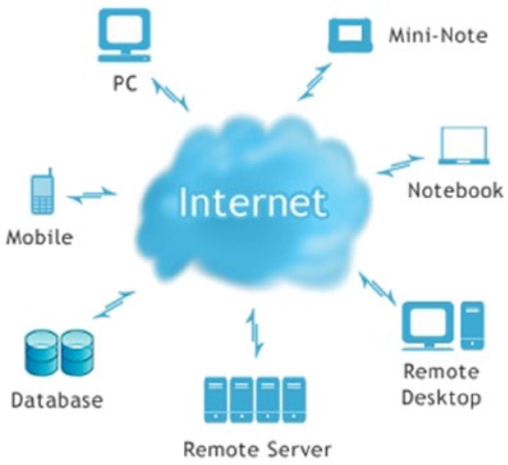


Figure 1-3. *The cloud was a metaphor for the Internet*

Figure 1-4. *Dragons were a metaphor for the uncharted parts of the map*

By moving your SharePoint infrastructure or software into the cloud, there is a danger that too much trust is placed in the platform provider to automatically take care of all the high availability and disaster recovery options. They do, in most cases, provide excellent tools to manage your infrastructure, but you must still know how to use them. The truth is the final responsibility still rests with the owner of the data to understand the options and choose the best ones for their needs and budget.

Instead of some nice, healthy fear, there is dangerous complacency that comes from a reluctance to take control of the infrastructure. It is easier just to assume someone else it taking care of it. I take it, dear reader, that you bought this book because you don't want to get swallowed up by the great chewing complacency.

Why Is Infrastructure Moving to the Cloud?

We live in a more connected world. Wi-Fi, smartphones, tablets, notebooks, and laptops allow workers to be more mobile and connection options more plentiful. People can access so much and communicate so easily through the Internet that they now expect to be able to access their work data from any location with any device with the same ease.

Another major factor in the arrival of the cloud for businesses is technologies like virtualization and cheap hardware, which allow for the commoditization of resources to the point that they are like any other utility, such as power, water, or gas. SharePoint needs a lot of hardware and capacity. The standard build is three farms: Development, Testing, and Production. SharePoint also requires a lot of software and licenses if you want, for example, three web front-end servers, two application servers, and a SQL cluster in each farm.

SharePoint Online (SPO) makes paying for access much simpler. There is no need for a large upfront investment in hardware, software, and licenses. Organizations can just sign on and pay monthly per user. They can even invite users from outside their network; this just requires a LiveID account like Hotmail or an existing Office 365 account. This makes collaborating beyond your network with partners or customers so much simpler. This also makes starting small and adding users gradually much easier—and the costs of user licenses up front much lower. It is much easier to remove user licenses, too, because each user has to re-authenticate once every 30 days; thus, once the 30-day license has expired, you no longer have to pay if you don't want to. There's no requirement to buy and configure a number of servers and work out what server and software licenses you will need. This has always been an overly complex and arcane art and any simplification here is very welcome. It is true there are still a range of user licenses to choose from, but the options are clearer and it's easier to identify what you want.

Licenses are also priced differently. They are now per user and not per device with SharePoint Online. The Client Access Licenses (CALs) for SharePoint 2013 are per device, so if you access from home, office, and mobile, you need three licenses, in theory, which is not something most organizations plan for. With SPO, a user can connect with up to five devices but it counts as only one device—a more realistic approach in this connected age and something Microsoft is counting on by building integration with Lync, SharePoint, and Exchange into its Windows Mobile platform.

In theory, administrators will no longer need to install patches. Of course, Microsoft will still be patching the platform, but this is no longer an administrative burden in the hands of the client to test and update servers on premises. Single sign-on does require ADFS and Directory Synchronization on premises as well as Office Professional Plus and Office 365 Desktop, and these will likely still require patching. This is still less than maintaining a stack of SharePoint and SQL servers.

Another important change is that the concept of versions becomes less significant. Online applications are gradually improving. People don't run different versions of Gmail, for example. So there's no longer a need to upgrade to the latest version of SharePoint every two or three years to get the latest features, maintain compatibility with other software, and keep the product supported.

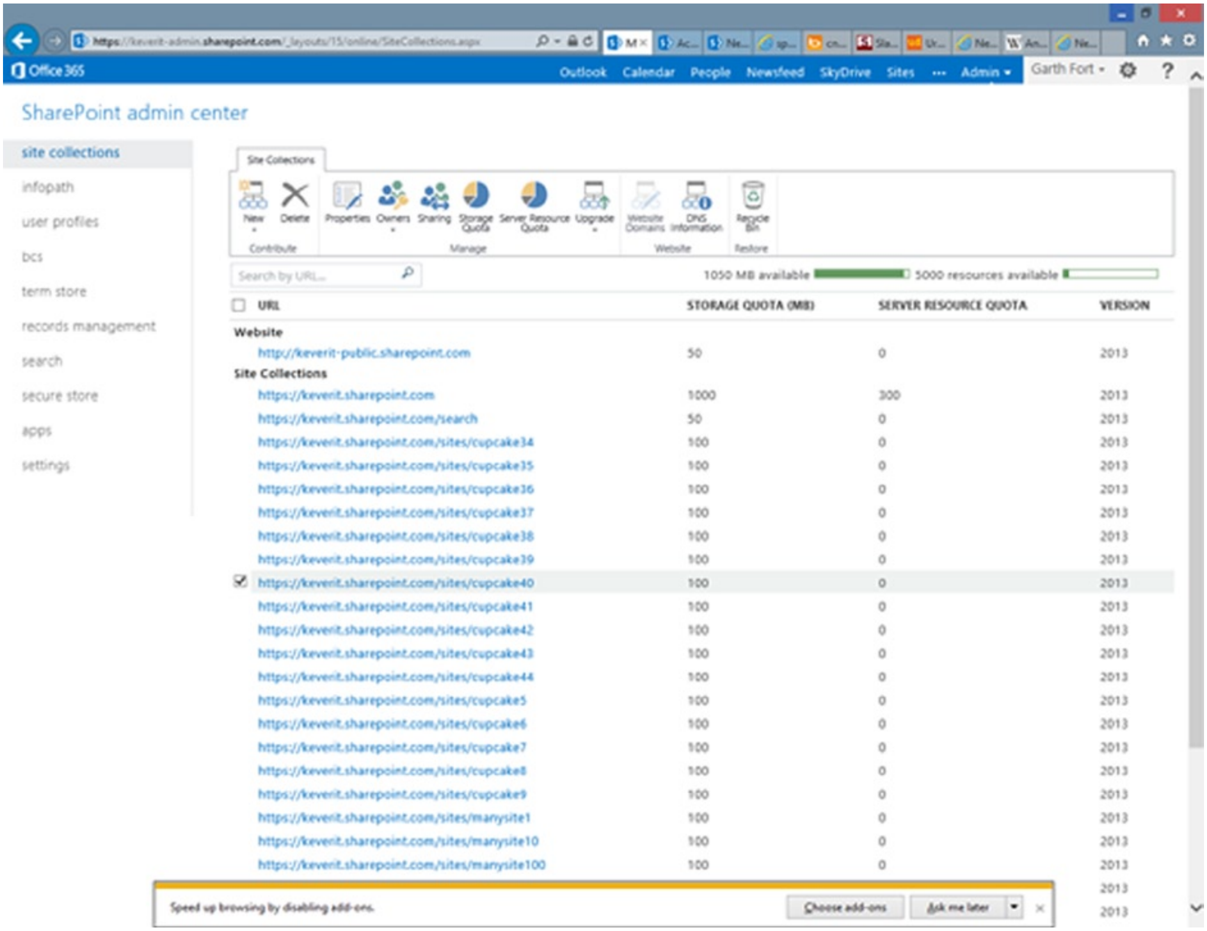
The disadvantages of SPO all come from the fact that it is a shared environment with other tenants and with that you have less direct access to the infrastructure. This means the powerful scripting tool PowerShell, which is used to automate administrative tasks and change farm settings, is not allowed. Like renting a flat in a house, you have less ability to change things that could affect all the tenants, so the Central Administration site is not accessible and farm-level solutions are not allowed.

Will SharePoint Administrators Become Extinct?

No, but they will have to evolve. SharePoint was once more like the manufacturing industry. It was about making and managing real things: servers and software. Now it is more like a service industry. The emphasis is delivering user satisfaction. Meeting the businesses requirements was always the purpose of technology, but now the emphasis is doing it more quickly and directly by listening to user requirements and helping then use SharePoint to meet them. SharePoint is more like Word than Exchange. Its value comes from the users and administrators knowing how to use it. There is also still resource management, user management, and quota management, as well as meeting branding requirements and declarative workflows through SharePoint Designer. Finally, through sandboxed solutions (though deprecated) there is still the ability to develop compiled code solutions through Visual Studio. There is also the SharePoint Apps Store where enhanced features can be purchased and installed. These will require an administrator to manage.

SharePoint Is a Complicated Beast

Moving to the cloud makes the technical aspects of setting up SharePoint much simpler. SharePoint 2013 is more complex than SharePoint 2010 or Microsoft Office SharePoint Server (MOSS) 2007 were—and SPS 2003 and SPS 2001. This is mainly because the services now run as applications in their own right. As a result, setting up a SharePoint farm can mean planning for more than 15 databases and learning how to configure at least as many services. SharePoint Online takes away some of that complexity, since there are only a limited number of services you can access. This is because the SharePoint Online infrastructure is standardized for all tenants; see Figure 1-5. It is the “Any customer can have a car painted any color that he wants so long as it is black” approach employed by Henry Ford.



☐ URL	STORAGE QUOTA (MB)	SERVER RESOURCE QUOTA	VERSION
Website			
http://keverit-public.sharepoint.com	50	0	2013
Site Collections			
https://keverit.sharepoint.com	1000	300	2013
https://keverit.sharepoint.com/search	50	0	2013
https://keverit.sharepoint.com/sites/cupcake34	100	0	2013
https://keverit.sharepoint.com/sites/cupcake35	100	0	2013
https://keverit.sharepoint.com/sites/cupcake36	100	0	2013
https://keverit.sharepoint.com/sites/cupcake37	100	0	2013
https://keverit.sharepoint.com/sites/cupcake38	100	0	2013
https://keverit.sharepoint.com/sites/cupcake39	100	0	2013
☑ https://keverit.sharepoint.com/sites/cupcake40	100	0	2013
https://keverit.sharepoint.com/sites/cupcake41	100	0	2013
https://keverit.sharepoint.com/sites/cupcake42	100	0	2013
https://keverit.sharepoint.com/sites/cupcake43	100	0	2013
https://keverit.sharepoint.com/sites/cupcake44	100	0	2013
https://keverit.sharepoint.com/sites/cupcake5	100	0	2013
https://keverit.sharepoint.com/sites/cupcake6	100	0	2013
https://keverit.sharepoint.com/sites/cupcake7	100	0	2013
https://keverit.sharepoint.com/sites/cupcake8	100	0	2013
https://keverit.sharepoint.com/sites/cupcake9	100	0	2013
https://keverit.sharepoint.com/sites/manysite1	100	0	2013
https://keverit.sharepoint.com/sites/manysite10	100	0	2013
https://keverit.sharepoint.com/sites/manysite100	100	0	2013
			2013
			2013

Figure 1-5. *SharePoint Online’s standard administration interface*

The onus is still on you to understand the options that are available and the pros and cons of the different decisions you can make. These decisions will affect the integrity of your system. This book is about helping you understand all the options so you can make an informed choice.

Practical Steps to Avoid Disaster

> *The art of losing isn't hard to master;*
> *so many things seem filled with the intent*
> *to be lost that their loss is no disaster*
>
> —"One Art" by Elizabeth Bishop

Do you think your SharePoint implementation is filled with the intent to become a disaster? This is a message you need to communicate to the people who can take the steps to avert it. Real action and real responsibility must be taken. There has to be consensus, too; it might be tempting to see this as a lone hero's struggle for recognition, but the way to avert disaster will require team cooperation. The art of losing is not hard to master. The art of success is much more difficult, but here are some practical steps.

What Role Will You Play?

It's important to consider what your role will be, both in the setup and later, if there is a disaster, in the cleanup. There are a number of different roles and responsibilities assigned to different people during the creating of a SharePoint deployment, and there are a number of roles and responsibilities that must be assigned in the event of a disaster. Make sure this does not happen in an ad hoc way.

If disaster happens, everyone is initially implicated and there will be an investigation to find out the causes and who, if anyone, has a part in the blame. Which role will you take if your ship founders?

- An engineer/administrator working to mitigate the disaster?
- A passenger/user, panicking and not helping?
- Someone who saw that the ship was sinking and only worked to save themselves?
- Or a hero who labored selflessly to save what they could?

Stakeholders and Strategy

Ownership of SharePoint is complex. Content is owned by users in the business. Sites and site collections are managed by site owners. Farms or tenancies are owned by IT staff. Branding and the look and feel are owned by the Marketing department. People invariably want ownership but not the responsibility that comes with it. They especially don't want accountability when things go wrong. So the first step in having good high availability and disaster recovery practices is establishing who is accountable for what.

SharePoint ownership is fundamentally a collaborative process. Creating good high availability and disaster recovery practices requires planning and commitment up front. This will also lead to a shared solution that will help the organization meet their top-level goals. Someone must lead this process and lead by example: take responsibility and be accountable, not just own the process long enough to take credit for it before moving on to something else that allows them to advance their career. That, dear reader, is you. If not, why not? If no one is taking responsibility for good high availability and disaster recovery practices in your organization, you should do it. Not just for fame or glory, but because you are a professional and a grown-up.

The next step is to create a cross-functional group that meets every month or six weeks to initially reach a consensus of what the organization is trying to accomplish. Without a shared understanding, you will not gain a shared commitment to the solution. Without a shared commitment, the good high availability and disaster recovery practices will eventually fail. This group must meet every two to three months to revisit the good high availability and disaster recovery practices to ensure they are still current to requirements.

Dependencies

This group must focus on the following key dependencies:

- Reliability will be a key indicator of success for the new SharePoint solution. It drives user adoption and maintains the valuable data already compiled by users.
- It will require a commitment from the owners of the infrastructure, the information architecture, and the content to ensure practices stay current. Keep responsibility with the owners, not one level above, as this disconnection leads to mistakes.
- Content will require the application of metadata and content types, which are part of the information architecture, to leverage the benefits within SharePoint to identify content that may be so valuable it needs its own high availability and disaster recovery policy apart from the rest of the content.
- Good high availability and disaster recovery practices cost time and money, but they cost a lot less than zero availability and zero disaster recovery. Without these, there can't be good practices.

Clear Measurements of Success: Reporting, Analysis, and Prevention

Simply measuring success by the fact that there has not been a disaster yet is not enough. My contention is that reporting of near misses by observers is an established error-reduction technique in many industries and organizations and should be applied to the management of SharePoint systems. Error logs only tell us so much. The majority of problems are those that people are aware of every day. There must be a place to record these observations. This must be a log that tracks these near misses in a transparent way: everyone should be able to read the log. It shouldn't be required to say who recorded the observation but it's better if people do so. This should begin to foster an environment of trust. It must be clear to everyone that the purpose of the log is not to apportion blame but to prevent security breaches or system outages. Table 1-1 shows a simple example form that could be maintained as a SharePoint list.

Table 1-1. *A Near Miss Log Form for Your SharePoint Farm*

Date/Time	Part of SharePoint	Type	Description	Contributing factors	Learning point	Action taken	People involved

The following are some questions that need to be discussed and reviewed with the cross-functional group. These are the kinds of questions that, when applied in real world situations, can help spot and address any problems sooner.

- Are there any patterns and trends?
- Is everyone competent to carry out the role they are assigned to do?
- Could this near miss combine with other near misses to create a chain of problems that could create an actual system failure?
- What resources are needed to address this problem?

Applied Scenario: The System Is Slowing Down

Ingenious Solutions Ireland has a problem with SharePoint. Users are reporting performance is slower than usual. They have called Support but Support hasn't been able to help. Likewise, the Infrastructure team says they will look into it but nothing further happens.

Members of Management notice, too, and eventually one of them asks the head of Infrastructure about the problem. At this point, the SQL server guy says the SQL servers are at capacity and he's been complaining about this for months but no one has done anything.

Infrastructure tells Management that users are putting too much into SharePoint and it's the user's responsibility to remove unneeded content. There are no policies around what has to be archived or deleted or even what should be put into SharePoint in the first place. The shared drive is full, too, so people have been putting everything and anything into SharePoint.

Content owners respond by saying all the content is necessary and it's the Infrastructure team's job to provide more space. Management realizes they need to buy more capacity but they didn't plan for this; when they see the rising cost, they do nothing in order to avoid having to tell Upper Management that they made a mistake and more money is needed to keep SharePoint going.

Eventually, the morale of the company is affected. Then one day the whole farm stops working. The redundant front end, application, and SQL servers are irrelevant because the problem was not caused by software or hardware. It was caused by people not taking responsibility for their part of the solution. After the disaster, SharePoint is offline for almost a week as some emergency freeing up of space is done to get things going again.

Upper Management hires a consultancy company to fix the problem. They quickly work out the real problem, which is no sense of responsibility or ownership. However, they convince Management that the solution is to buy a new, expensive, trendy content management solution and pay them to support it. They only take responsibility for setting up the new system and some basic training. Thus the process starts all over again.

The Solution

Invariably, there is a cycle of failure, and spotting it is the first step. In this example, the shared drives filled up, so SharePoint was used as a solution. Then it filled up and the latest trendy solution was brought in instead.

Upper Management should get the cross-functional group together with a representative from the following groups:

- Content owners
- Site administrators
- Management
- Support
- Infrastructure

They need to work together to reach a consensus on what went wrong. They should communicate without apportioning blame to each other. Upper Management should provide a person to guide and own this process, keep it on track, and keep everyone involved until a course of action is set.

This should be a constructive process where the conclusion is that each member takes responsibility for their contribution to the problem.

- Content owners take responsibility for not uploading content to SharePoint unless it is there for the distinct business processes agreed upon. If these are not known, they must be defined. They also take responsibility for regularly deleting content that is no longer needed. This has the benefit of keeping search results relevant, makes navigation faster, and makes the system less cluttered.
- Site administrators take responsibility for maintaining quotas on sites plus archiving and deleting sites that are no longer needed.
- Management takes responsibility for providing enough money to prevent the system from running out of resources. They also take responsibility for providing resources for training for users, site administrators, and support staff.

- Support takes responsibility for making users and site administrators vigilant in deleting unneeded content. If content needs to be removed but archived, they report this to Infrastructure.
- Infrastructure takes responsibility for having a system in place to archive content and for using that system. They also regularly monitor capacity; if they reach a specific target, say 25% of storage capacity, they report this to Management.

What Is Upper Management's Responsibility?

The role of Upper Management is to maintain ownership of this whole process. They can't simply subcontract it to an external consultancy. If they find they don't understand the technology involved, they need to get themselves the necessary training to understand the issues involved. Technology is now vital to the brand, morale, and financial health of every organization. It is too important to be just ignored in the hope that things will just continue on as they were. The problem is they will—and this will cost money and even good staff, who may leave.

After these steps have been completed, the near-miss log should be implemented. Anyone in the organization can contribute to it, but it should be reviewed weekly by management and all points should be discussed at the monthly SharePoint cross-functional group meeting. It is Upper Management's job to make sure these meetings take place and that the near misses are addressed. In this example, the symptom was slow performance, and the causes were multiple: poor content retention policies, lack of training, and lack of capacity or budget. The cause was actually lack of attention to the importance of IT processes in the business. The cure was Upper Management taking ownership of creating the processes need to prevent problems like this from happening again.

Technology Is Just a Tool

Did you notice I mentioned almost nothing technical in this example? I didn't go into detail about the structure of the SharePoint farm, how many front-end servers, application servers, or even SQL servers in the cluster. I didn't talk about the advantages of mirroring versus clustering or combining the two. I didn't mention stretched farms, DR farms, SQL backups, or tape backup. This is because none of that would have made any difference. It is assumed that whoever created the SharePoint farm initially followed Microsoft's well-documented processes on creating highly available SharePoint farms, or hired someone who knew how to specify the hardware and software and then install and configure a SharePoint farm. The problem was something harder to measure—and what can't be measured can't be managed. In the example, the farm did not fail because of technology; it failed because of people.

There is a tendency to see high availability and disaster recovery as purely technical areas. In my opinion, the technology is the simplest part to manage (even though it still takes a great deal of work to master it—and I don't think anyone ever fully can). Many companies sell the idea that you can buy a magic solution to the problems of high availability and disaster recovery, that their skills or tools to monitor or backup the farm will mean you will never lose access or data.

At the root of the problem is the fact that SharePoint itself is just a tool, like a hammer, a car, or a telephone. Microsoft sells it, but it's up to you to work out what to use it for and, more importantly, how to manage it so that it keeps working and meeting your needs.

Microsoft designs, manufactures, and distributes SharePoint. Partners sell it. Third parties provide add-ons to it. Consultancies install, configure, and support it. They also develop and design custom functionality. Training companies show you how the default functionality works. But in the end, it's up to the owners of the tool to use it to its full potential and maintain it in a way that it remains useful.

Applied Scenario: It's Never Simple

Examples tend to be simple and clear. But the real world they try to illustrate is complex and unclear. Complexity and a lack of clarity is the main problem we all face in attempting to solve the high availability and disaster recovery problems of most companies. If a problem is simple to frame, it's usually simple to solve. Here is a scenario involving the kind of messy situation that leads to poor high availability and disaster recovery decisions.

Super Structure is an IaaS company. Their customer, Fancy Flowers, contracts them to design a highly available and recoverable SharePoint 2010 farm. Then they change their mind and ask for a SharePoint 2013 farm. Super Structure doesn't have SharePoint 2013 experience, so they subcontract an external consultancy, Clever Consultants, to provide the expertise. They also subcontract Dashing Development to provide custom coding.

After a long and exhaustive process, the solution architecture is agreed upon. This takes time because 15 to 20 people (representatives from the four companies) are directly involved. There are multiple meetings and documents. A detailed design is drawn up and the servers are all installed and configured.

Two months later a project manager from Fancy Flowers notices that her idea for high availability and disaster recovery—a stretched farm—is not in the solution architecture. In the minutes of the meeting, she sees that everyone agreed that this was the way to go. Actually, the Solution Architect from Clever Consultants argued that a stretched farm across two data centers would only provide high availability up to a point because the SAN was still in the first data center. If that data center went down, the farm would go down too. Super Structure argued that the SLA that Fancy Flowers paid for was their highest level and this would take too long to recover from. The actual solution in the solution architecture was for a disaster recovery farm in the second data center. But Fancy Flowers insisted that this be marked as "part of phase 2" and so it was described in the solution architecture but not actually implemented in the detailed design.

At this late stage, the Fancy Flowers project manager balks and says that is not what they agreed. She doesn't think the cost of the disaster recovery farm is necessary, despite her lack of knowledge of SharePoint, and she insists things must be done her way.

Dashing Development stays out of this because Fancy Flowers is their customer and they don't want to lose the business of designing custom branding and web parts for them.

The discussion now whirls between 15 to 20 people as to what the SLA means and how this should be delivered. Super Structure offers a third option: log shipping and moving the staging farm to the second data center to double as a DR farm. This will mean uninstalling and reinstalling this farm from scratch as all the accounts and machine names include the name of the data center. Also, log shipping will mean further capacity in the network to store the logs.

Someone brings up mirroring and there is much debate about what databases can, should, and shouldn't mirror in SharePoint 2013. The discussions reach a stalemate as no one seems to be willing or able to make the final decision. In the end, they do nothing. Eventually, the data center is destroyed when a local river bursts its banks. Without a proper disaster recovery plan, everyone sues everyone else and it cost them all a lot of money while they all continue to believe they were right all along.

Some Terminology

Before I go on, I'll explain some of the terms that are useful to know in the context of HA and DR.

- *SAN*: Storage that can be used by SQL or Windows.
- *Latency*: How long it takes data to travel from one server to another. Low latency (1 millisecond) is ideal.
- *Data center*: A building with lots of servers in it. Primary and secondary data centers are normally less than 100 km apart because fiber optic cable starts to degrade in efficiency at lengths longer than that, thereby increasing the latency.
- *Replication*: Copying data between farms.

- *Mirroring*: Making a copy of data almost instantly, like a mirror. Referred to as synchronous, which means “at the same time.” Because it has to be done constantly, it requires lots of system resources.
- *Log shipping*: This means backing up the SQL data to a file server and then restoring it to another SQL server. Referred to asynchronous because the copying is not done instantly. Typically the log backup might be done at 6 p.m., but the restore would be done six hours later. This is because it takes perhaps two hours to back up the SQL databases to a file server, two hours to copy them to a file server on in the other data center, and two hours to restore the data to the farm.
- *Hot, warm, and cold standby*: If a standby system can be operational in minutes or less, it’s referred to as hot. If it takes several minutes or hours, it’s referred to as warm. If it takes many hours or days, it’s referred to as cold. These are not exact terms.

Summary of the Options

Later in Chapter 4, I will go into the relative merits of these choices in more detail; I will also explain how to implement them. For now, here is a summary of the options in this scenario.

Option 1: Log Shipping/Mirroring

This was the option presented by Super Structure. As an IaaS company, they have experience with Windows and SQL Server but not with SharePoint, so they proposed that some of the databases could be mirrored or log shipped to the secondary data center and that the farm intended for the staging of new code could double as the disaster recovery farm. This plan includes separate web servers, application servers, and SQL servers in the two data centers. Only some of the databases can be log shipped with SharePoint, and since they are different farms, the configuration databases are not replicated (I will go into more detail on this in Chapter 4. In Figure 1-6, you can see that some content and service databases are being replicated and that the servers in the same farm are in different data centers. Table 1-2 covers the pros and cons of this approach.

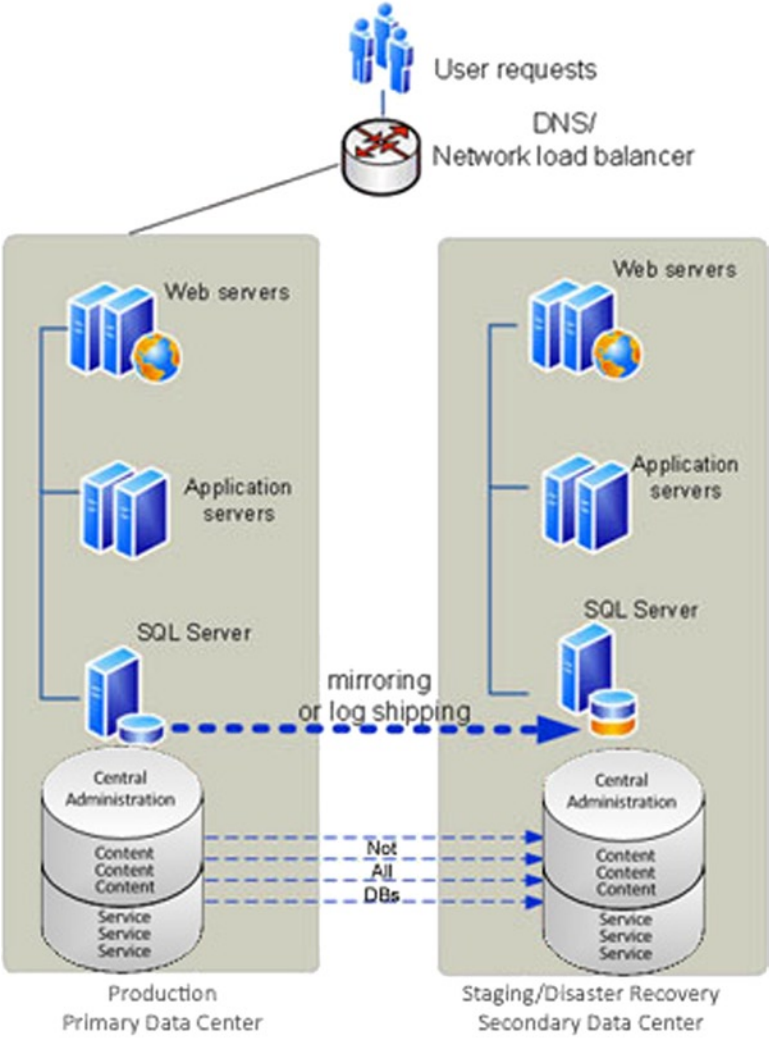


Figure 1-6. *A SharePoint farm with mirroring or log shipping*

Table 1-2. *The Pros and Cons of a SharePoint Farm with Mirroring or Log Shipping*

Pros	Cons
Standard way to provide HA and DR for SQL Server.	SAN not replicated to secondary data center. SAN mirroring/ replication costs millions of dollars.
Asynchronous: providing cold standby availability in hours or days.	Logs are not copied in real time between the principal and the mirror servers, so no negative effects on performance.
A compromise of cost versus benefits.	File server space required to hold logs during copy to DR farm.

Option 2: Stretched Farm

Figure 1-7 shows the architecture Fancy Flowers wanted. They did not specify mirroring specifically as they didn't know the difference between log shipping and mirroring, but to give highest availability of the SQL layer, it would be a good idea. There are web front-end servers, application servers, and SQL servers in the primary and secondary data centers. All are in the same farm. However, there is an important detail not represented in this diagram that Super Structure didn't explain to Fancy Flowers: the SAN. The storage for the whole farm is still in the primary data center. So if that data center became unavailable, it would mean a lot of stretching was done for no real benefit. Replicating the SAN is possible, but it's also a very expensive option—much more expensive than having a separate disaster recovery farm in the secondary data center. But Fancy Flowers is set on this idea and isn't budging. Table 1-3 represents the pros and cons of this approach.

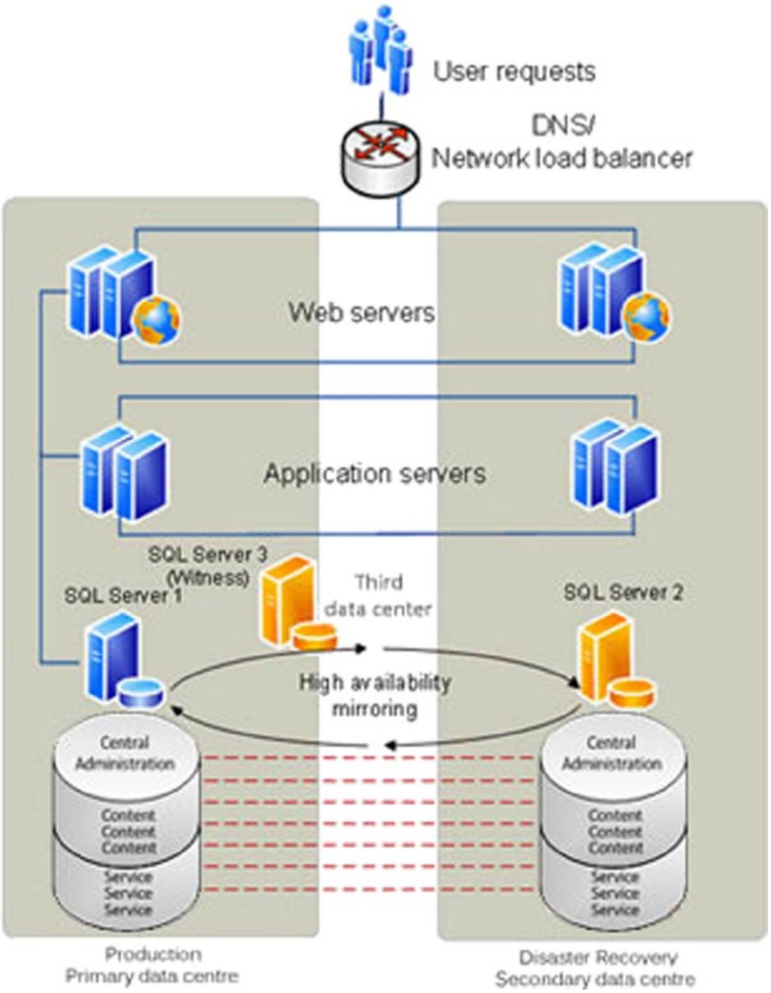


Figure 1-7. *A SharePoint stretched farm with mirroring*

***Table 1-3.** The Pros and Cons of a SharePoint Stretched Farm*

Pros	Cons
In SLA, it corresponds to highest level of availability.	SAN replicated to secondary data center. SAN mirroring/replication costs millions of dollars.
Provides high availability and disaster recovery.	Mirroring is expensive in terms of system resources.
Synchronous, thus providing hot standby availability in seconds or minutes.	Support for stretched farms was deprecated in SharePoint 2013 but has now been resupported in limited circumstances. There must be less than 1 millisecond latency between SQL Server and the front-end Web servers in one direction, and at least 1 gigabit per second bandwidth.

Option 3: Disaster Recovery Farm

Option 3, shown in Figure 1-8, is almost exactly the same as option 1 except here the disaster recovery farm is only used for that purpose. It is not used for the staging of custom development. This is what was proposed by Clever Consultants but rejected by Fancy Flowers because of the additional cost. However, it is less expensive than SAN replication and certainly better than no disaster recovery at all. Table 1-4 shows the pros and cons of this option.

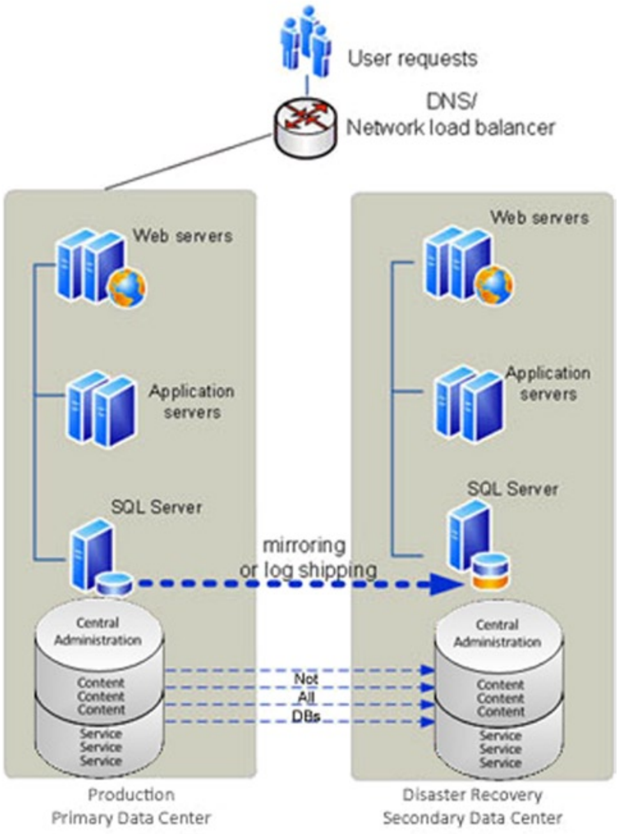


***Figure 1-8.** A disaster recovery farm*

Table 1-4. *The Pros and Cons of a SharePoint Combined Staging/Disaster Recovery Farm*

Pros	Cons
In SLA, it corresponds to high level of availability.	SAN replicated to secondary data center. SAN mirroring/replication costs millions of dollars.
Provides high availability and disaster recovery.	Mirroring is expensive in terms of system resources.
Asynchronous, thus providing warm standby availability in minutes or hours.	More costly than no DR farm. Second farm and capacity paid for even when not used.
Same level of performance after failover.	File server space required to hold logs during copy to DR farm.
Database layer is asynchronously log shipped.	Logs are not copied in real time between the principal and the mirror servers, so no negative effects on performance.
No dependency on constant connectivity between data centers.	Requires secondary farm to be maintained/patched to keep same as primary.

The Solution

This is a typical scenario because of the multiple people involved and the multiple technical options. Here the failure came about because there were too many people involved and no one person with enough knowledge or authority to make the decision. There were four parties with different motivations and no cooperation between them. It would be easy to blame the disaster on the river, but in fact it was poor project management that really caused the disaster.

It's not uncommon for organizations to subcontract to other companies because they lack the expertise to make the technical decisions. Here are some details on the four players involved:

- Fancy Flowers: They wanted the most secure solution but also the cheapest. They didn't accept that disaster recovery is expensive and that designing your own solution if you don't understand the technology is a recipe for disaster. Being the client, they had veto power on all decisions; also they reversed the decision on the solution architecture after it was agreed, which caused chaos.

- Super Structure: They had the infrastructure expertise, but SharePoint requires specialist knowledge, which they lacked. When they had to deliver an SLA to Fancy Flowers, they fell back on what they knew: SQL server log shipping as the solution.

- Clever Consultants: They were stuck in the middle. They had responsibility for the solution architecture, but they lacked authority to push their solution through. In the end, they compromised on their initial recommendation of a disaster recovery farm to get the solution signed off.

- Dashing Development: They stayed neutral through all this and managed not to get any of the blame. Their goal was to stay in good graces with Fancy Flowers, so they decided to neither help nor hinder.