A service-oriented architecture (SOA) is a vendor, product, or technology independent pattern that enables application components in a network to obtain services from other components. You can take advantage of the Microsoft Integration Stack to implement your SOA strategy smoothly and cost effectively.

SOA Patterns with BizTalk Server 2013 and Microsoft Azure, Second Edition provides an overview of Microsoft Integration technologies, including BizTalk Server, and demonstrates how you can leverage these technologies to implement a successful SOA.

Starting with an introduction to BizTalk Server and WCF, you will learn about RESTful services, JSON, and Azure Service Bus, and how to apply SOA principles to creating BizTalk solutions. Moving on, you will learn how to implement versioning in endpoints, orchestrations, and schemas. Finally, you will discover the usage of the ESB Toolkit and extend the connectivity of BizTalk Server applications to Microsoft Azure.

Who this book is written for
If you are a developer who has been tasked with building service-oriented BizTalk Server solutions, this book is for you. It will help you to envision an enterprise solution and implement the software blueprint.

What you will learn from this book
- Understand how to implement SOA with BizTalk Server and the Azure platform
- Consume and expose WCF services effectively via the use of Service Bus Relays and RESTful services
- Implement effective schema design, including an introduction to various schema design patterns
- Exploit various message exchange/endpoint patterns including request-response, fire and forget, and client callbacks
- Leverage orchestration design patterns that maximize flexibility and reuse
- Future-proof your BizTalk Server artifacts using well thought out versioning strategies
- Build loosely-coupled BizTalk applications using the ESB Toolkit
- Take a peek at API Apps, Logic Apps, and Azure API Management
In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 13 'New SOA Capabilities in BizTalk Server 2013 – Azure Hybrid Patterns'
- A synopsis of the book’s content
- More information on SOA Patterns with BizTalk Server 2013 and Microsoft Azure Second Edition
About the Authors

**Richard Seroter** is the VP of Product for the CenturyLink platform, and a Microsoft MVP for application integration. He is also an instructor for the developer-centric training company Pluralsight, the lead InfoQ editor for cloud computing, and the author of multiple books on application integration strategies. As the VP of Product for CenturyLink, he is responsible for product strategy, sprint planning, and community contribution. He also leads an expert team of product owners and analysts there. He maintains a blog ([https://seroter.wordpress.com/](https://seroter.wordpress.com/)) that he updates regularly on the topics of architecture and solution design and can be found on Twitter at @rseroter.

**Mark Brimble** is a solution integration architect with over 17 years of experience in the field of designing, building, implementing, and supporting integration solutions. He has experience in integrating Windows, Unix, and AS/400 platforms, SQL Server, SAP, Oracle, Microsoft CRM, and other web service technologies. He also has experience of eGate (Java CAPS), SSIS, BizTalk Server, MuleSoft, Boomi, and Flow integration products. His main interest at the moment is to help people choose the best architecture patterns. He has been active in the integration community for many years and posts regularly on his blog at [https://connectedpawns.wordpress.com/](https://connectedpawns.wordpress.com/). He also contributes to BizTalk Map Documenter ([http://biztalkmapdoc.codeplex.com/](http://biztalkmapdoc.codeplex.com/)) and is the coordinator for BizTalk 2013 Documenter ([https://biztalk2013documenter.codeplex.com/](https://biztalk2013documenter.codeplex.com/)). He can be found on Twitter at @BrimbleMark.
**Johann Cooper** is an integration specialist who currently holds the role of a principal integration consultant at Datacom, New Zealand, and focuses on Microsoft Stack. He is well-versed with the usage of BizTalk Server, WCF, and RESTful services. He recently delved into the cloud space and successfully delivered projects that leverage the Microsoft Azure platform. He is a keen advocate of following and defining best practices, and is a big believer in automated testing.

He is the creator and curator of the *BizTalk BRE Pipeline Framework* CodePlex project, the author of the blog *Adventures inside the Message Box*, and is the author of the whitepaper *The A-Y of running BizTalk Server in Microsoft Azure*. He is also part of the Microsoft P-TSP program that is tasked with the goal of promoting the usage of BizTalk Server and the Microsoft Azure platform in New Zealand.
Colin Dijkgraaf started dabbling in programming as a teenager in the days when the Internet was still called FidoNet and the modem speed was around 2400 baud. After getting a bachelor’s of commerce degree in information systems, he did various different IT-related jobs, including image banking and digital imaging. He finally started full-time development for Datacom in 2000. In 2004, he first came across BizTalk while developing the frontend for an interchange to process invoices and purchase orders for a large New Zealand company and also worked on some maps. He has become a full-time developer of BizTalk since then and also a codeveloper of BizTalk Server 2013 Documenter.

Mahindra Morar has been working in the IT sector since 1997, developing Windows and website enterprise applications. In the last 8 years, he has focused primarily on integrating systems as a principal integration consultant. Having come from an electronics engineering background, he is able to use this knowledge to design solutions that integrate wetware, software, and hardware. He has worked in many industries, including manufacturing, financial institutions, insurance, retail/wholesale, and power utilities. His areas of interest include exploring new technologies and deciding how to use them in the world of integration.

You can view his blog at https://connectedcircuits.wordpress.com.

The last 6 months have been a great learning experience in writing this book. I now truly appreciate the hard work and dedication authors endure to complete a book. I take my hat off to you all.
Preface

Repeat after me: SOA is something you do, not something you buy.

– David Linthicum

This may seem an odd quote to use when beginning a book about employing a particular product to facilitate the implementation of a service-oriented architecture (SOA). However, I think it sets the tone.

I prefer to define SOA as an architectural discipline based on loosely-coupled, autonomous chunks of business functionality, which can be used to construct composite applications.

This is how the first edition of this book started, and like so many words in that book, they are timeless. As I updated chapters, I was continuously struck by how little had changed. We also added new chapters, Chapter 4, REST and JSON Support in BizTalk Server 2013, Chapter 5, Azure BizTalk Services, and Chapter 6, Azure Service Bus to cover some of the newer technologies that give us more ways to implement SOA. For sure, there are newer technologies and we are embracing the brave new world of the cloud, but the underlying patterns that we use in integration do not change. For example, the words in Chapter 7, Planning Service-oriented BizTalk Solutions just as true today as they were seven years ago. SOA might not be a trendy word anymore, but call it what you will, the pattern does not go away.

– Mark Brimble, June 2015
From writing desktop and web-based Line of Business (LOB) applications in the early 2000, I find integration is the adhesive that connects all these disparate systems together. Without integration, I wonder how many businesses would survive in today's world where information exchange is essential. This book describes how BizTalk Server and Microsoft Azure create the adhesive that bonds these systems together.

When writing the chapters on REST and hybrid systems, I pondered how the Internet has grown to become the conduit of connecting systems and devices together. The trend now seems to be all about providing RESTful services and hosting them in the cloud. With this in mind, new integration patterns have emerged, which are discussed in this book.

– Mahindra Morar, June 2015

In the information technology industry, if you aren't constantly learning, then you are falling behind due to the technology and methodology constantly changing. Some examples of this changing landscape include the move toward cloud-based services and REST web services, both of these are covered in this updated book.

I'm lucky enough to be part of a team of BizTalk developers with whom I can share and receive ideas and concepts, and we can learn from each other. Another source of these ideas and concepts come from books like this where others have shared their expertise. So, I hope this book helps you to keep learning and growing.

– Colin Dijkgraaf, June 2015

BizTalk Server is equivalent to a Swiss army knife in that there are so many options available to solve integration problems. Like a weapon, it is possible to attack your problems with brute force or finesse. Typically, the end result seems similar, but the amount of effort expended is vastly different, not to mention the amount of cleanup required after the exercise.

When I was still in the early stages of my journey toward becoming a seasoned BizTalk developer, I came across the first edition of this book, and it opened my eyes to the fact that there is a right way to do integration, and an expensive way to do integration, even if the costs are not instantly evident. The differentiating factors between these two end results typically come down to applying the right integration patterns and making early architectural decisions that will guide you down a path toward building efficient solutions.
Now, I'm not just talking about the 40,000 feet in the air high level types of architectural decisions, which are obviously very important, but the low level detail as well. Having an understanding of the inner workings of the toolsets you are working with as well as an appreciation for how and when to apply various integration principles and patterns can make or break a solution. These patterns and principles are timeless, and will serve you well regardless what technology you are using to solve your integration problems, but having a keen understanding of their relevance to a given product empowers you even more.

The first edition of this book would have already given you great insight into the inner workings of BizTalk Server and how to apply SOA principles to the platform. This updated edition will take things further, firstly by extending the existing material from a BizTalk Server perspective with all the new insights that have been gleaned in the last few years. Moreover, we have also explored some previously untrodden territory by exploring how SOA and BizTalk Server can be used to build effective solutions leveraging RESTful architectures as well as by extending the reach of the integration platform through the use of Azure Service Bus. Finally, we end the book by giving you a taste of the newest technologies to be released by Microsoft, which will hopefully inspire you to get your hands dirty playing with these new platforms.

Thank you for reading our book, and I hope you enjoy reading it as much we enjoyed writing it.

– Johann Cooper, June 2015

What this book covers

Chapter 1, Building BizTalk Server 2013 Applications, looks at what exactly BizTalk Server is, reviews the core architecture of the application, and shows you how to build an end-to-end solution.

Chapter 2, Windows Communication Foundation Primer, looks at the problems that WCF attempts to solve, and how to actually build and host WCF services.

Chapter 3, Using WCF Services in BizTalk Server 2013, builds a number of common scenarios using BizTalk and WCF services.

Chapter 4, REST and JSON Support in BizTalk Server 2013, covers the fundamentals of REST-based services and the BizTalk WCF-WebHttp adapter.
Chapter 5, *Azure BizTalk Services*, looks at one of the newer integration offerings from Microsoft, reviews how it works, and shows how to build an end-to-end solution. Although this platform has been superseded by Azure Logic Apps while we were writing this book, we decided to keep this chapter for historical completeness.

Chapter 6, *Azure Service Bus*, discusses the different types of services available and the characteristics of each type. You will also learn about creating Azure Topics and Subscriptions.

Chapter 7, *Planning Service-oriented BizTalk Solutions*, investigates exactly what a service-oriented BizTalk solution looks like. What types of services you should expose. How you can exchange messages through the BizTalk bus. We'll answer these questions and many more at this stage of the book.

Chapter 8, *Schema and Endpoint Patterns*, explores various patterns for building schemas and endpoints, and discusses what scenarios each pattern is applicable to.

Chapter 9, *Asynchronous Communication Patterns*, looks at how to take advantage of asynchronous messaging to build robust service-oriented solutions. We'll also cover the tricky concept of providing acknowledgements or results to clients that call services in a fire-and-forget fashion.

Chapter 10, *Orchestration Patterns*, explores advanced patterns used to build loosely-coupled orchestrations that cater for advanced functionality without compromising flexibility.

Chapter 11, *Versioning Patterns*, covers proven methods to version your BizTalk components, ensuring that you don't introduce changes that will cause unintended disruption.

Chapter 12, *Frameworks and Tools*, introduces you to a variety of tools and frameworks that can be leveraged to support your SOA aspirations.

Chapter 13, *New SOA Capabilities in BizTalk Server 2013 – Azure Hybrid Patterns*, covers the pros and cons of hybrid solutions. This chapter also describes the different integration patterns available when connecting on-premise resources to the resources hosted in the cloud.

Chapter 14, *What’s New and What’s Next?*, will touch upon some of the latest offerings from Microsoft, cover how they are relevant to you, and give you a taste of how to implement solutions with these platforms.
Once you have eliminated the impossible, whatever remains, however improbable, must be the truth.

– Spock, Star Trek

With the advent of cloud computing, there is now more demand for bridging on-premises applications to cloud-hosted applications and services.

A hybrid cloud solution is when applications or services are spread across two or more private, public, or community clouds.

A private cloud is an infrastructure dedicated to a single organization. There are two variations on this, on-premises private cloud and an externally hosted private cloud. An on-premises private cloud is hosted within an organization’s own data center, while an externally hosted private cloud is located in a public data center with resources dedicated solely to your organization.

A public cloud belongs to a service provider who hosts the infrastructure and is generally available to the public. Customers who use this, typically share the same infrastructure resources with other customers with limited configuration and security options.

A community cloud is shared infrastructure between several organizations that is governed and managed by all participants.
With hybrid solutions, there are many obstacles to overcome, such as limited bandwidth, secure channels, and data solvency, to name a few.

In this chapter we will discuss the following:

- Advantages of hybrid solutions
- What SOA patterns hybrid solutions tend to use
- Special security considerations
- Monitoring assets on the ground and on the cloud

**Advantages of a hybrid solution**

Creating a hybrid solution provides many advantages compared to a totally on-premises application or service. Some of the key benefits of shifting to a hybrid solution are as follows:

**Reduced operating costs**

By paying only for what you actually use, you can reduce the operating costs involved. Extra resources may be purchased at any time and then relinquished once peak loads have subsided.

For this to work effectively, you would host resource-intensive processing in the cloud, which can be scaled when required and back down again after the results of the processing are returned to the on-premises application.

**Freeing up on-premises infrastructure resources**

On-premises infrastructure resources can be made redundant or freed up by moving systems into the cloud, thus reducing licensing costs.

**Allowing burst capacity**

Most cloud infrastructures provide the capability to dynamically scale resources up horizontally and vertically as demand increases or down as demand tapers off. This offers a big advantage when dealing with burst type loads.
Improved service levels
Hybrid solutions provide better availability service levels, as the cloud components can easily be scaled up as demand increases. Also, cloud infrastructure tends to be more resilient to failure.

On the other hand, cloud-based integration services lack inbuilt persistence points and retry logic. When you design an integration solution for the cloud, you should provide persistence points within a process flow and retry logic for connections to on-premises resources. This will deliver a more resilient solution.

Reduced capital expenditures
This is one of the biggest advantages of cloud computing: moving from a capital expenditure model (CAPEX) to an operational expense model (OPEX).

Testing and trialing innovative ideas can easily be accomplished without having to outlay any infrastructure requirements. Normally, you just purchase extra cloud resources when required and have them up and running within minutes.

Improved system availability and disaster recovery
Cloud infrastructure is built on a robust architecture, providing resiliency and redundancy.

Most cloud resources are normally duplicated to other resource groups in a totally separate rack. Geo-replication is also provided as an option for total disaster recovery.

Wider audience reach
By hosting some of the components of a system on the cloud, it allows a wider reach of consumers as these services are not obscured by corporate firewalls.

In these scenarios, components that handle sensitive information normally reside on-premises and public-facing APIs are hosted in the cloud, where a secure channel connects the distributed components.
Disadvantages of a hybrid solution
Implementing a hybrid solution does have some disadvantages, namely:

- **Security**: The hybrid solution is solely reliant on the provider to provide best practices in protecting your data from unauthorized access.
- **Provider dependency**: Normally called vendor lock-in. This is when it becomes difficult to move your resources from one vendor to another. An example that could make it difficult would be the amount of data to transfer to another vendor.
- **Technical issues**: Diagnosing and fault finding system errors that have components distributed across several networks.

SOA patterns used in hybrid solutions
A typical hybrid solution will have resources distributed between cloud infrastructure and on-premises resources behind a corporate firewall.

Before deciding on a hybrid solution, there are several key considerations that must be taken into account. Cloud integration solutions should provide the following benefits:

- **Scalability**: Allows the system to accommodate more demand by adding more resources horizontally or vertically. Horizontal scaling (or scaling out) lets you adjust the number of instances while vertical scaling (or scaling up) allows you to change the size of the instance.
• **Elasticity**: The ability to dynamically grow and shrink resources, depending on the demand.
• **Security**: Managing access to resources in a secure manner.
• **Robustness**: Handling failures, network interruptions, and so on in a graceful manner.
• **Flexibility**: To vertically or horizontally scale the resources.
• **Configurability**: Providing tools or APIs to configure the environment.

There are two types of cloud integration patterns:

• **Cloud-to-cloud**: Commonly referred to as SaaS integration. This is where applications and services connect to each other via the cloud.
• **Cloud-to-enterprise**: Also commonly known as hybrid integration. This is where there is a secure connection between cloud-based and on-premises services/applications.

On-premise applications tend to have low network latency and very seldom encounter network connectivity issues; cloud services, on the other hand, tend to have higher network latency and network transients. Without the proper business logic in place to handle connectivity issues, there is a risk of losing information.

There are many scenarios where data cannot be moved into the cloud and must be accessible by applications that are hosted externally. The most common way of connectivity was via a secure connection through a firewall and NATs.

Now there are other alternatives using BizTalk Services, as described in the following sections.

**BizTalk Services Hybrid Connection**

BizTalk Services Hybrid Connections provides a convenient way to connect Azure Websites and Azure Mobile Services to on-premises resources behind your firewall. It is one of the fastest ways to build hybrid applications as it does not require a VPN gateway or firewall changes to allow inbound traffic. It works by installing a connection agent service onto an on-premises server that connects to the local resource. The connection agent then listens for any connections from the Hybrid Connection in Azure.

The most obvious scenario where you would use a Hybrid Connection is when a resource is not designed as a Microsoft WCF service, in which case you might want to employ Service Bus Relays instead. External devices simply access the on-premises resources in the same manner as if they are connecting to a local resource.
A good example for using a Hybrid Connection is if you wish to connect directly to an enterprise (on-premises) database, as shown in the following diagram:

---

**Azure Relay Services**

Azure Relay Service is very similar to Hybrid Connections. However, the main difference is Relay Services only supports WCF applications that use the WCF Relay bindings.

As described in Chapter 6, *Azure Service Bus*, Relay Service is another option for providing connectivity for hybrid solutions, by delegating listening for incoming requests to the cloud-hosted Service Bus.

In the following diagram, we see a typical scenario where an on-premises WCF service needs to be exposed externally. By using an Azure Service Bus Relay, the external consumers connect to the endpoint hosted on the cloud, which relays the requests to the on-premises service:
BizTalk Adapter Service

This adapter enables your cloud application to communicate with on-premises LOB systems.

The following LOB systems are currently supported:

- Microsoft SQL Server
- Oracle Database
- Oracle E-Business Suite
- SAP
- Siebel eBusiness Applications

The Adapter Service consists of a WCF LOB target adapter that is installed on an IIS server, located on-premises. One end of this adapter connects directly to your resource, and the other end connects to an Azure Service Bus, using your service namespace.

When a message arrives at your BizTalk Service in Azure, it is placed on Service Bus and is monitored by the on-premises LOB adapter. The adapter then executes the CRUD operation and returns the response via the Service Bus.
New SOA Capabilities in BizTalk Server 2013 – Azure Hybrid Patterns

BizTalk Server Adapters

BizTalk 2013 now includes new adapters that allow on-premises services and applications to extend to the cloud.

These new adapters have been mentioned in the previous chapters, but I will list them here again as these adapters may typically be used for hybrid solutions also. Remember that these adapters are available for both BizTalk receive locations and send ports.

The WCF-BasicHttpRelay and NetTcpRelay adapters

The WCF-BasicHttpRelay and NetTcpRelay adapters allow sending and receiving messages through the Azure Relay Service using BizTalk as the service broker between an on-premises application and a cloud-based application.

In the following diagram, the enterprise applications expose their APIs using one of BizTalk's WCF Relay adapters. The WCF Relay adapter makes an initial outbound connection to the Service Bus Relay, which establishes the connection to the cloud. Whenever the tablet device requires resources from the applications residing on-premises, it sends a request to the Service Bus Relay, which forwards the message to BizTalk's WCF Relay adapter:
This scenario is a typical hybrid pattern to employ when you need to expose on-premises resources as WCF services, using BizTalk as a service broker.

**The SB-Messaging adapter**

This adapter provides first-class integration with Azure Service Bus Queues and Topics. Using this type of connectivity allows many cloud-based integration patterns to be utilized (publish/subscribe, disconnected clients, load management, and so on).

In the following solution, the SB-Messaging adapter monitors the specified Azure Service Bus Topic for any new messages by polling the endpoint. When a new message arrives on the Queue from the tablet device or any other client application, BizTalk will read it off the queue for processing:

[Diagram showing Service bus topic, Corporate Firewall, BizTalk IIS, CRM, and ERP]

**The BizTalk WCF-WebHttp adapter**

The BizTalk WCF-WebHttp adapter allows developers to support RESTful API type calls that are lightweight. Many SaaS providers are adopting REST as the default protocol to use in order to consume their services.

The WCF-WebHttp adapter offers the option to expose your on-premises applications as RESTful services by using BizTalk and IIS to host the services. This is an especially useful capability to expose RESTful APIs for enterprise applications that do not provide any web service capability.
Using the WCF-WebHttp adapter in a practical example was discussed in *Chapter 4, REST and JSON Support in BizTalk Server 2013*.

---

**Azure SQL Data Sync**

In some scenarios, it is beneficial to store a subset of data on the cloud in order to reduce network latency between a cloud application and on-premises databases.

This, however, brings forth another problem with data synchronization between on-premises and cloud-hosted databases. By using the SQL Azure Data Sync service, which is built upon the Microsoft Sync Framework, the databases can be scheduled to synchronize at intervals of between 5 minutes to a month without writing any code.

Synchronization may be set up in a singular or bi-directional manner between any combination of cloud and enterprise databases. You also have the option to synchronize down to tables and columns within a database.

If you require near real-time data synchronization, you may want to consider AlwaysOn availability groups instead.
Microsoft Azure Caching

Although this section is not part of integrating applications, it is still worth a mention as Azure Caching plays an important role in hybrid solutions to provide scalability.

Microsoft Azure Caching provides a robust caching mechanism in scenarios where networking connectivity is limited in bandwidth or is subjected to networking transients between cloud and on-premises applications.

However, to use this feature, your application would be required to make use of caching in the data access layers unless caching was already implemented from the initial design.
Database sharding
This is another pattern worth discussing when developing hybrid solutions. Database sharding is horizontally partitioning data in a table. It improves scalability with storing and accessing vast amounts of data.

In a hybrid solution, the on-premises table schemas would be replicated in a cloud hosted database. Data intended for cloud resources would then be stored on the cloud partition.

Special security considerations
Security concerns are one of the primary hindrances to large-scale adoption of cloud computing today. When you move applications to the cloud, you cannot rely on traditional security measures as you are no longer in control. You tend to gain a sense of security because the networking and infrastructure hardware is managed by world-class security practices.

As with all applications, the greater the surface area that is exposed, the greater the chances of exposing a security vulnerability. Most cloud technologies and services are exposed as endpoints rather than in-memory communication pathways. This increases the surface area that may potentially be exploited.

With hybrid data storage, it is normally good practice to only place less critical data on the cloud and leave all sensitive data on-premises, where you have total control over access.

If storing sensitive data on the cloud, you should consider some form of encryption mechanism.

Providing a first-class authentication and authorization mechanism is the first line of defense for your services. This should be based on industry-proven standards, such as two-factor authentication or token based security, where authentication is managed by another identity management service.

Auditing all service calls and authentication requests is another form of defense against attacks. In some cases, it is possible to be alerted when malformed requests are actively being sent.
Monitoring assets on the ground and on the cloud

As with any service or application, monitoring should be one of the top priorities in the overall solution design. It is important to monitor how well the overall system is functioning and how quickly issues are resolved when they arise. Without proper monitoring and alerting in place, there is no means to check the health of a system and to manage any service level agreements (SLA) that are in place.

Monitoring hybrid solutions is not an easy task as components of the system are now distributed across different locations and connected by networks that may vary in bandwidth and reliability. You also have fewer configuration options for cloud-hosted components and are reliant on the data center to provide their own monitoring tools.

With Microsoft Azure, you can use Azure Diagnostics to gather performance and diagnostic information about your components running on each infrastructure node. Azure Diagnostics has been specifically designed for cloud applications to minimize the impact on performance when collecting diagnostic data.

It is highly configurable by specifying what information to collect, whether it be data from performance counters, event logs, IIS logs, or crash dumps.


Another recently released monitoring service from the Microsoft Azure team is called Application Insights. It works by installing an SDK package into your application to send telemetry data to the Azure Portal. The SDK is available to support multiple platforms and development languages. These applications or services do not need to be deployed onto the Azure platform.

More information about this service can be found here at http://azure.microsoft.com/en-us/services/application-insights/.

There are also many other third-party products available for monitoring private, public, and hybrid clouds.

To provide a centralized view of the captured diagnostic data from your hybrid solution, there are a few options, as described next.
If your organization is currently using System Center Operation Manager on-premises, there is a System Monitoring Pack available for Azure applications. The default configuration monitors the state of the hosted service and roles, ASP.NET performance counters, memory utilization, and processor performance.

With System Center Operations Manager, you can configure alerts to be raised when various counters have exceeded the specified thresholds and automate tasks to be executed. For example, the ability to spin up additional web instances if the web response times are becoming too long.

Another option is to configure Azure Diagnostics to persist the captured data to blob storage, as shown in the following diagram. Then, at regular intervals, copy the data stored in blob storage to a centralized database situated on-premises. Applications running on-premises would also write diagnostic information to this same database. A reporting tool, such as Microsoft Reporting Services, would then be used to send alerts and reports on the captured data:

Ensure that the diagnostic information repository is secure, as it may yield security information and information about the internal structure of your system. All communication between the Azure storage and the on-premises applications should be over a secured channel, such as HTTPS.
Handling scalability, availability, and performance

Organizations tend to move applications to the cloud because of cost savings and the scalability factor. In doing so, an additional level of management complexity is added due to resources being shared by other tenants who may introduce risk by affecting your applications. Some of this risk may be mitigated by providing multiple instances of the application or by opting for premium tier offerings, which provide dedicated resources rather than shared resources.

For a mission-critical application to handle high demands, these factors must be considered carefully during design and must be developed from the beginning, instead of being added to the solution afterwards.

Components of a solution that are quite chatty should be kept in close proximity to each other and not distributed across networks.

Scalability

Using cache in solutions is one of the most significant options you can use to enhance the scalability of an application. Typically, cache should be used when frequently querying static data from persistent storage, such as databases and blob storage. However, you must consider the amount of required cache memory and the expiration policy. For a hybrid solution, you can use Azure cache to minimize traffic between cloud and on-premises resources for slow-changing data, instead of querying the on-premises resource for every request.

Using multiple Azure Worker roles is another option to use when processing asynchronous tasks. More roles may be bought online to handle the extra demand. In a hybrid scenario, messages would be sent to an Azure Service Bus Queue from on-premises applications and then picked up by several Azure Worker roles to process the data asynchronously.

Availability

To ensure high availability, a cloud provider may use a combination of load balancing, partitioning, and resource management.

Most cloud providers offer redundancy by creating three copies of storage resources on different rack systems in the same data center. This ensures high availability of data if a hardware device fails. When this does occur, the Fabric Controller will transparently switch to another copy of the storage resource.
Geo-replication is also available on some cloud providers where, data in a primary location is asynchronously replicated to a secondary location. There is no dependency between these locations other than the transferring of data.

Microsoft recommends having a minimum of two compute instances to ensure high availability of services. This is because the virtual machine that hosts the compute instance may be rebooted by the Fabric Controller after a planned maintenance event. The Fabric Controller will never place two compute instances on the same virtual machine. This is to ensure both instances never get rebooted at the same time.

### Performance

One of the major hurdles with hybrid solutions is network performance between the cloud and on-premises network infrastructures. Applications and services migrated to the cloud may need to be optimized for cloud architecture. This may include a significant redesign of the solution.

Placement of the data is very important with hybrid solutions. Tightly coupled data may not allow the application to perform well if it needs to be distributed across domains.

Most cloud providers offer effective management tools that allow performance monitoring of applications. Using these tools offers the ability to adjust the resources dynamically.

### Summary

In this chapter, we've discussed hybrid solutions and architectures. We have learned the benefits of moving to a hybrid solution and also discussed some of the issues that are inherently associated with cloud infrastructures.

We looked at different options to connect on-premises and cloud-hosted resources using BizTalk Server and other Azure products.

We also discussed how to design scalable cloud applications using caching and multiple compute instances. Another important point that we touched upon was how to monitor hybrid solutions using System Center Operations Manager and how to leverage Azure Diagnostics and Application Insights.

In the next chapter, we shall take a look at some of the new features in BizTalk Server 2013 R2, Azure and beyond.
Where to buy this book

You can buy SOA Patterns with BizTalk Server 2013 and Microsoft Azure Second Edition from the Packt Publishing website.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

Click here for ordering and shipping details.