Identifying Workloads for the Cloud
Identifying workloads to move to the cloud can be tricky. You have dozens or hundreds of apps running in your organization, and now that you’re investing in the cloud, you’re tempted to move as many of them up there as fast as possible. As you’ll see in the examples below, cloud computing is indeed a good fit for many common workloads.

But remember the old saw, “You can lead a horse to water, but you can’t make it drink.” It turns out you can lead an app to the cloud, but it doesn’t always make it run better, so you’ll also see how to identify apps that are better left in your data center in their current architecture.

### 3 Quick Criteria for a Good Fit

Let’s start with three rules of thumb for what to put into the cloud:

1. **Unpredictable load or potential for explosive growth.** Whenever your app faces outward, it has the potential to be wildly popular. Social games, eCommerce sites, blogs and software-as-a-service (SaaS) products fall into this category. If you release the next Farmville™ and your traffic spikes, you can scale up and down in the cloud according to demand.

2. **Partial utilization** When traffic fluctuates – say with daily cycles of playing or shopping, or with occasional, compute-intensive batch processing – you can spin up extra servers in the cloud during the peaks and spin them down afterwards.

3. **Easy parallelization** Applications like media streaming can be scaled horizontally and are generally a good use case for the cloud, because they scale out rather than up.

Finally, keep in mind the ideal of cloud computing as a way of using multiple resource pools – public cloud, private cloud, hybrid, your internal data center – not choosing one over the others. RightScale lets you see and manage all of them through one interface and set of tools.
This brief covers the characteristics of three ideal workloads that RightScale manages in the cloud:

1. **Scalable Web Apps**
2. **Batch Processing**
3. **Disaster Recovery**

It depicts the working configurations of actual RightScale customers and ends with a list of workloads that are not a good fit for the cloud.

---

### Scalable Web Apps

**Characteristics**

Many Web apps have the potential for unpredictable spikes in traffic. Suddenly, you need to scale up your infrastructure very far and very fast to handle the extra load. After doing this a few times, you realize that you’d better automate it so that you don’t need to expend precious human talent on it.

**How the cloud helps**

Cloud computing gives you access to **unlimited resources on demand** – limited only by Amazon, RackSpace or whichever public cloud providers you’ve chosen – so as you experience these spikes in traffic, their entire resource pool is accessible to you. But because you don’t have to run all the infrastructure to cope with the peak load, you **pay for only the resources that you use**. Extra capacity costs you money only when you need it for spikes in traffic. But to truly utilize these resources on demand, you need to be able to scale automatically. By monitoring the load of your application, RightScale determines the need to expand your infrastructure with more servers and automatically adds new resources to the pool to handle the additional load. RightScale gives you the **automation tools and ease of operations** to launch new servers easily and connect them to the rest of your deployment, including your load balancers, databases and so forth.
**Architecture example**

The architecture below shows a typical web application. It includes a load balancing tier based on aiCache, an application tier running Python Django, a Memcached front-end for improved performance, and a master-slave PostgreSQL database tier.

With RightScale monitoring and alerting, the Django application tier auto-scales up and down. Application activity is monitored - for example, the level of CPU use and the number of incoming requests - and the optimal number of application server is always maintained. RightScripts or Chef recipes connect the application servers to the proper load balancers and databases automatically in real time.
Batch Processing

Characteristics
Batch processing and tasks like encoding and decoding data streams require concentrating a great deal of processing power on huge workloads. Time is often of the essence in these jobs, so the objective is to dedicate all available resources to them, then hand off the results to another process. The need for these jobs is not constant, and their fluctuating utilization cycle often means high usage today and low usage tomorrow; it’s not worth owning and maintaining all the infrastructure required for the spikes.

How the cloud helps
A cloud configuration can link metrics and auto-scaling to boost the computing capacity in a tier of encoding servers, for example, connected to a queue like RabbitMQ or even a database that stores jobs that need to be executed. The servers in the tier can encode jobs; then, based on the length of the queue and the load on each server, the system can automatically add servers to meet demand. So, if there are thousand jobs in the queue and each server can process only ten of them at a time, the system can spin up one hundred servers in the cloud to complete the task. Either a private or a public cloud (but especially a public cloud) puts virtually unlimited computing resources at your disposal. RightScale lets you bring up large numbers of servers fast and dedicate huge numbers of servers to the task, then retire them as soon as the peak has passed, so that you pay for only the resources you use.

Architecture example
The architecture on the following page is part of a mobile application that allows users to upload video. A batch process is needed to encode the video and serve it in different formats. It includes an auto-scaling Sorenson Squeeze array that performs the video encoding. This array scales based on the length of the job queue. A MySQL database maintains video meta-data.
2 Batch Processing

- Video encoding process auto-scales based on jobs in the queue

Diagram elements:
- DNS
- LOAD BALANCERS
- APP SERVERS
- QUEUE
- MASTER DB
- SLAVE DB
- RabbitMQ
- Cloud Files
- MySQL
Disaster Recovery

Characteristics
In case of a disaster or a problem in your production data center, disaster recovery depends on having the same set of resources at the ready in a different location and data center. You don’t want to buy and maintain an exact duplicate of your production-quality infrastructure when it’s going to sit idle most of the time, but when it is needed, you need to spin it up and cut over to it in a hurry to avoid downtime.

How the cloud helps
You can implement your disaster recovery plan in any public or private cloud, which gives you the advantage of geographic diversity. In the cloud, you can run a scaled-down version of your deployment at the disaster recovery site. For example, in the production data center and cloud, you might run your entire infrastructure – load balancers, servers, databases, etc. – then in the disaster recovery site you would run and pay for only a replicating slave database (or several databases if necessary to support your workload) to reduce costs. Then, when you need to execute your disaster recovery plan, you use RightScale ServerTemplates and associated RightScripts to quickly launch the application servers and load balancers in your deployment. Again, automation tools for smoother failover help you not only when you need to get all the bits of infrastructure up and talking to one another in a disaster, but they also make it easy for you to test and enhance your disaster recovery process regularly without a huge investment in hardware.

Architecture example
The architecture on the following page shows a warm DR environment build to withstand a cloud failure. The production environment (left) has been cloned into a different cloud (right) to provide high availability. Redundant load balancers and applications servers are staged and can be launched in a matter of minutes. Because a database can take much longer to come online, the redundant slave database is constantly running. Database replication and backups happen automatically using RightScripts. Automation tools switch environments in the event of an emergency, launching the staged servers, connecting the tiers, failing over the database, and switching over the DNS, connecting them to the load balancers and databases, failing over the master database to these slaves, and switching over DNS.
3 Disaster Recovery

1. Production servers are cloned into a warm DR environment, spanning one or many clouds.

2. Load balancers and application servers are staged to save money.

3. Databases automatically replicate.

4. Databases failover first to another zone and then to another cloud.

5. Database volumes are automatically backed up.
Workloads that Are Not a Good Fit for the Public Cloud

Characteristics
Sometimes the cloud isn’t the right answer. Here are some examples:

- **High-performance applications that demand a lot of disk I/O and network throughput**
  Some proprietary databases and the applications that run them require very high I/O and expect a consistent ability to read to and write from the disk systems. In general, the public cloud uses shared resources, so performance varies. Most of the time it doesn’t vary too greatly, but some databases are better off in a private cloud or a dedicated hybrid configuration. Sometimes the problem is in the application design, when developers create the app without an eye to launching it in the cloud and unintended side effects can ensue. And certain legacy and enterprise apps may simply not work in the cloud, so they need to stay on your own servers.

- **Applications that demand low latency over the network**
  Some databases – especially in analytics applications – require high-throughput replication and clustering, and it is still hard to deliver that throughput in the public cloud. Some organizations accustomed to high-performance network storage try to move their applications from network attached storage (NAS) or a NetApp appliance in a typical data center, and continue to use things like NFS storage in the cloud. But it’s hard to make an NFS server highly available in the cloud, and even then it won’t likely perform well. We see customers use S3 or Gluster for high-performance shared storage.

- **Specific hardware dependencies**
  Not all hardware is available in the public cloud, so configurations that depend on rare or obsolete equipment are better off staying in the data center.
Roll up your sleeves and try it. Now. Free.

You can try setting up and managing these workloads yourself with RightScale today.

Now that you’ve seen how much of your work RightScale can simplify, start working on your own configurations for:

1. **Scalable Web Apps**
2. **Batch Processing**
3. **Disaster Recovery**

Stay tuned for more webinar briefs. We’re making it so easy for you to get up and running with RightScale that you’ll wonder how you ever got along with it.

Try RightScale Free