AWS Intensive

WELCOME!

Develop Intelligence

John Kidd

Join Us in Making Learning Technology Easier

Our mission...

Over 16 years ago, we embarked on a journey to improve the world by making learning technology easy and accessible to everyone.

...impacts everyone daily.

And it's working. Today, we're known for delivering customized tech learning programs that drive innovation and transform organizations.

In fact, when you talk on the phone, watch a movie, connect with friends on social media, drive a car, fly on a plane, shop online, and order a latte with your mobile app, you are experiencing the impact of our solutions.

Over The Past Few Decades, We've Provided

Over **62,300,000** expert-led learning hours

In 2019 Alone, We Provided

Training to over **13,500** engineers

Over **120** active trainers, with an average of over two decades of experience each.

Programs in **30** countries

Develop Intelligence

Technologies we cover

Ruby	Jenkins	akka		ANGULAR	ANSIBLE	APACHE Spark
aws	Azure		cassandra	CHEF		docker
	GO		GraphQL	iOS	Java	JS
kafka		mongo DB	MySQL	node JS	puppet	python
		React Native	redis	Scala	spring	
		TS		AND MANY OTHER TRENDING TECHNOLOGIES		

World Class Practitioners

Develop Intelligence

250 best selling books authored

9+ years of training experience

Over 62 million practitioner led training hours

150 engagements speaking at industry conferences

Over 17 years of industry experience per instructor

125 certifications in leading technologies

95% instructor satisfaction

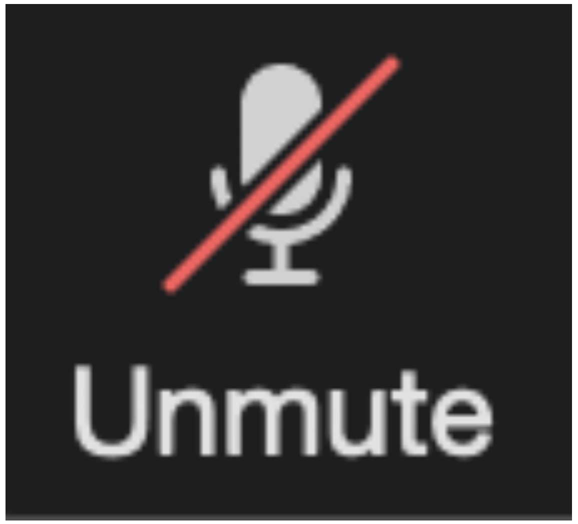
Note About Virtual Trainings

What we want

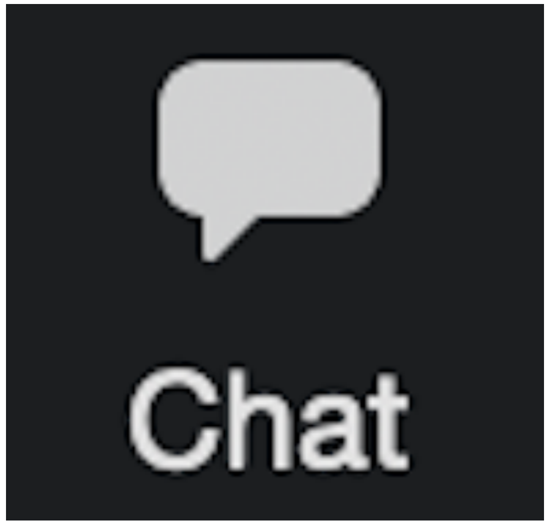
...what we've got

Virtual Training Expectations for You

Arrive on time / return on time



Mute unless speaking



Use chat or ask questions verbally

Virtual Training Expectations for Me

I pledge to:

- Make this as interesting and interactive as possible
- Ask questions in order to stimulate discussion
- Use whatever resources I have at hand to explain the material
- Try my best to manage verbal responses so that everyone who wants to speak can do so
- Use an on-screen timer for breaks so you know when to be back

Prerequisites

- In this course we'll be covering an introduction to AWS.
- Now- AWS offers upwards of 90 services to customers…so we will obviously not be going into deep dives on all of them (that's all we would talk about and we want to be able to do some practical work!)
- Our focus will be primarily on the commonly used parts of AWS to put up a robust, inexpensive, flexible architecture.
- The course will move between lectures and labs with the over-arching theme to create a single application.

Objectives

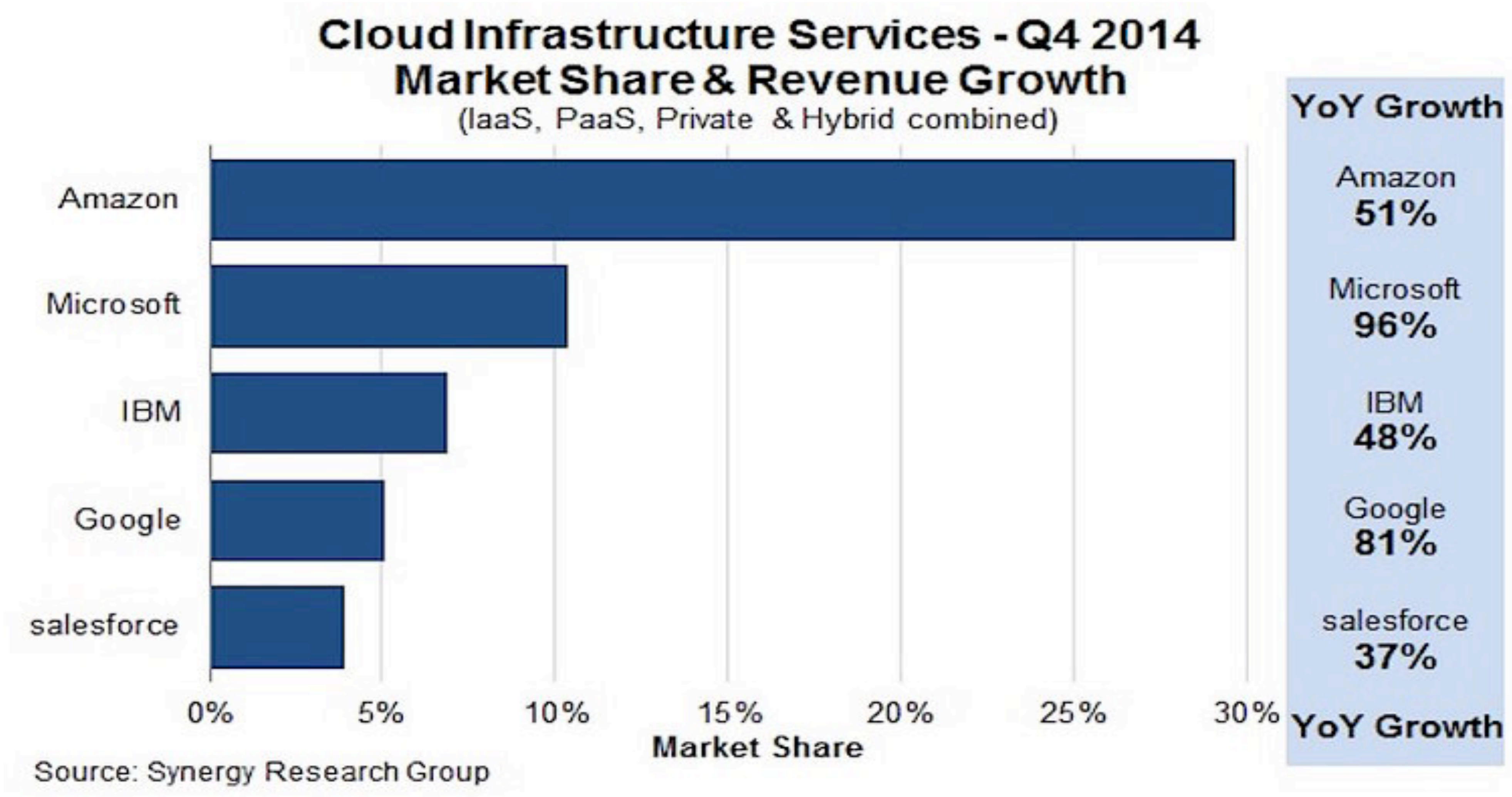
At the end of this course you will be able to:

- Create and manage users on AWS
- Deploy and manage different AWS systems
- Create and manage Virtual Private Clouds/Networks on AWS
- Create and manage virtual machines on AWS

Agenda

- Class schedule:
  - Pretty simple: Around 2.5 hours then 15 minute break broken into three parts:
    - Meet n' greet: 9:30 – 10:00 am
    - Lab Setup: 10:00 to 11:00am
    - Break: 11:00am to 11:15am
    - Lecture 1: 11:15 to 12:30 pm
    - Lunch : 12:30 to 1:30pm
    - Lab 1: 1:30pm to 3:00pm
    - Break: 3:00pm – 3:15pm
    - Lecture 2: 3:15pm to 4:30pm
  - Breaks:
    - As needed. Nothing formal (we're all adults!) but one rule:
      PLEASE do not return to your workstations and start "doing email" or I will NEVER SEE YOU AGAIN!

Why AWS?


Cloud Infrastructure Services - Q4 2014
Market Share & Revenue Growth
(IaaS, PaaS, Private & Hybrid combined)
Amazon
Microsoft
IBM
Google
salesforce
0%
5%
10%
15%
20%
25%
30%
Market Share
Source: Synergy Research Group
YoY Growth
Amazon 51%
Microsoft 96%
IBM 48%
Google 81%
salesforce 37%
YoY Growth

Structure of the Course / Takeaways

- We will need to have AWS accounts to run the course.
- Ideally, if you have an AWS account, that is the ideal.
- If not- please see me.

THANK YOU

Develop Intelligence

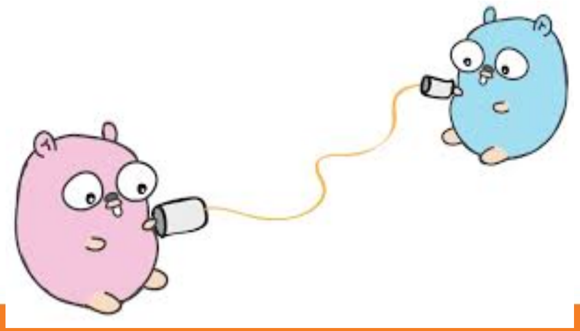
DevelopIntelligence

Systems Operations

Develop Intelligence

So what does an Dev Ops do?

- AWS Devops is a unique approach to Devops that most of us probably aren’t used to.
- The way that AWS Devops usually operate is around creating and maintaining networks
- It’s a lot more than the traditional, old school “let’s just keep making VMs for people”

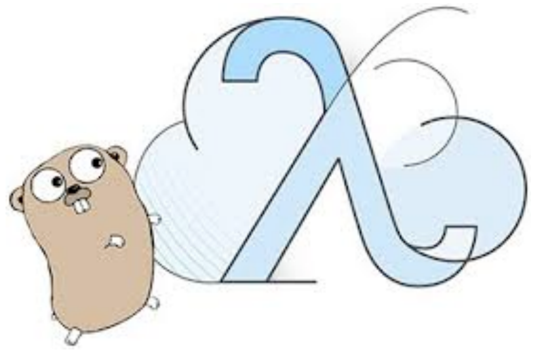


The AWS DEVOPS Role

- Things that the DEVOPS has in common with current DEVOPS rules
  - Availability: Obviously the devops is the one who gets called at 3am when the site goes down; as such site availability is one of the primary responsibility of the AWS Operations manager
  - Security: Sort of “everyone’s responsibility” (as we are frequently reminded)
  - Networking: Making things talk to each other
  - Environment management: Keeping everyone’s environments straight- including protecting production and making dev easy to access
  - Keeping Costs under control

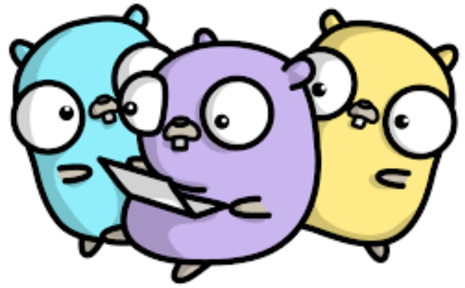
The AWS Devops role

- ”Doing DEVOPS” in AWS centers around managing demand vs time. Management of resources, in the traditional sense of “make this server work” is generally offloaded to third parties like Amazon AWS.
- Since resources are virtually unlimited (want a new server? Simple to create in seconds!) the major role of DEVOPS is now managing the existing resources and most importantly *not letting your environment get out of control*



The DEVOPS Role

- We're going to talk about this again and again- but in this new environment your job is less about keeping dying servers alive (we just shoot a dying server and recreate it; we don't waste time trying to keep it alive) and more about managing those resources
- Who gets what and when
- This will make you the most popular person at the company (as always)



DevelopIntelligence

The AWS Well Architected Framework

Develop Intelligence

Brief history around AWS

- Original idea was around an initial beta released in 2002 (called the Amazon.com Web Service that offered SOAP and XML interfaces for the Amazon product catalogue.
- In 2003, during an executive retreat at Jeff Bezos' house, the Amazon leadership team was asked to identify the core strengths of the company.
- What emerged from that was the infrastructure advantage that Amazon had over it's competition.

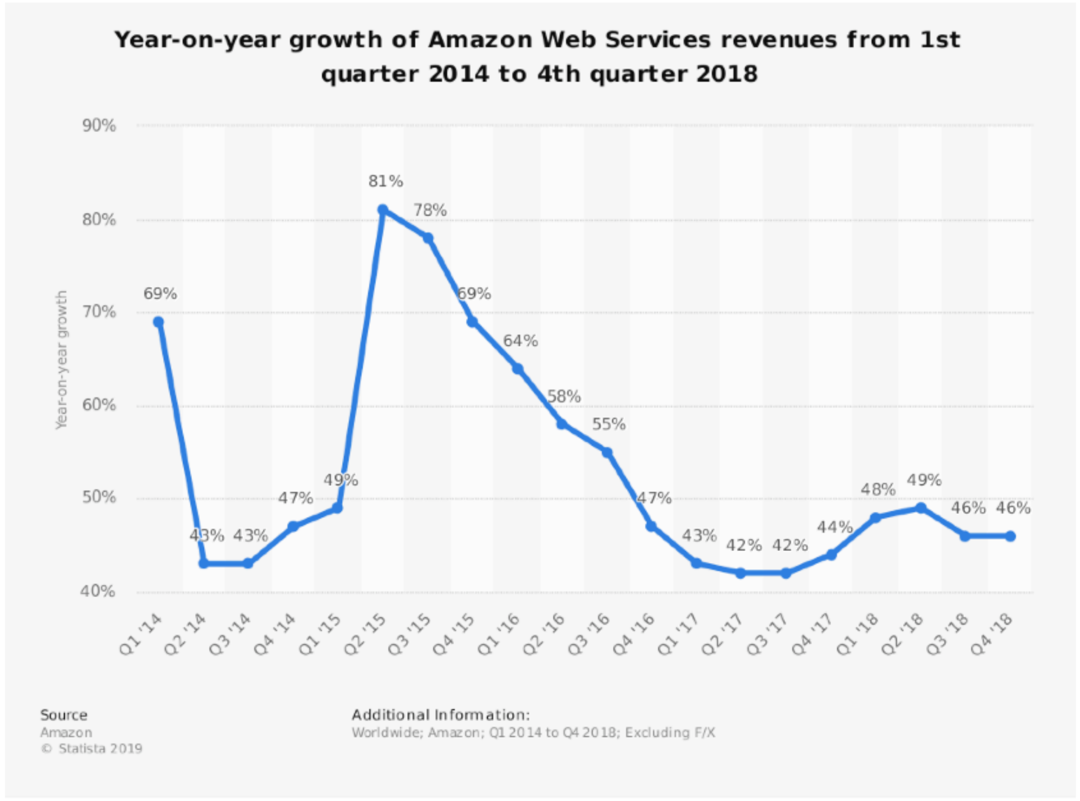
Brief history of AWS (continued)

- The initial AWS offering was launched on March 19, 2006 offering two services:
  - S3
  - EC2
  - (SQS followed soon after)
  - Elastic Block Store came after that
  - Then the Content Delivery Network known as Amazon Cloudfront

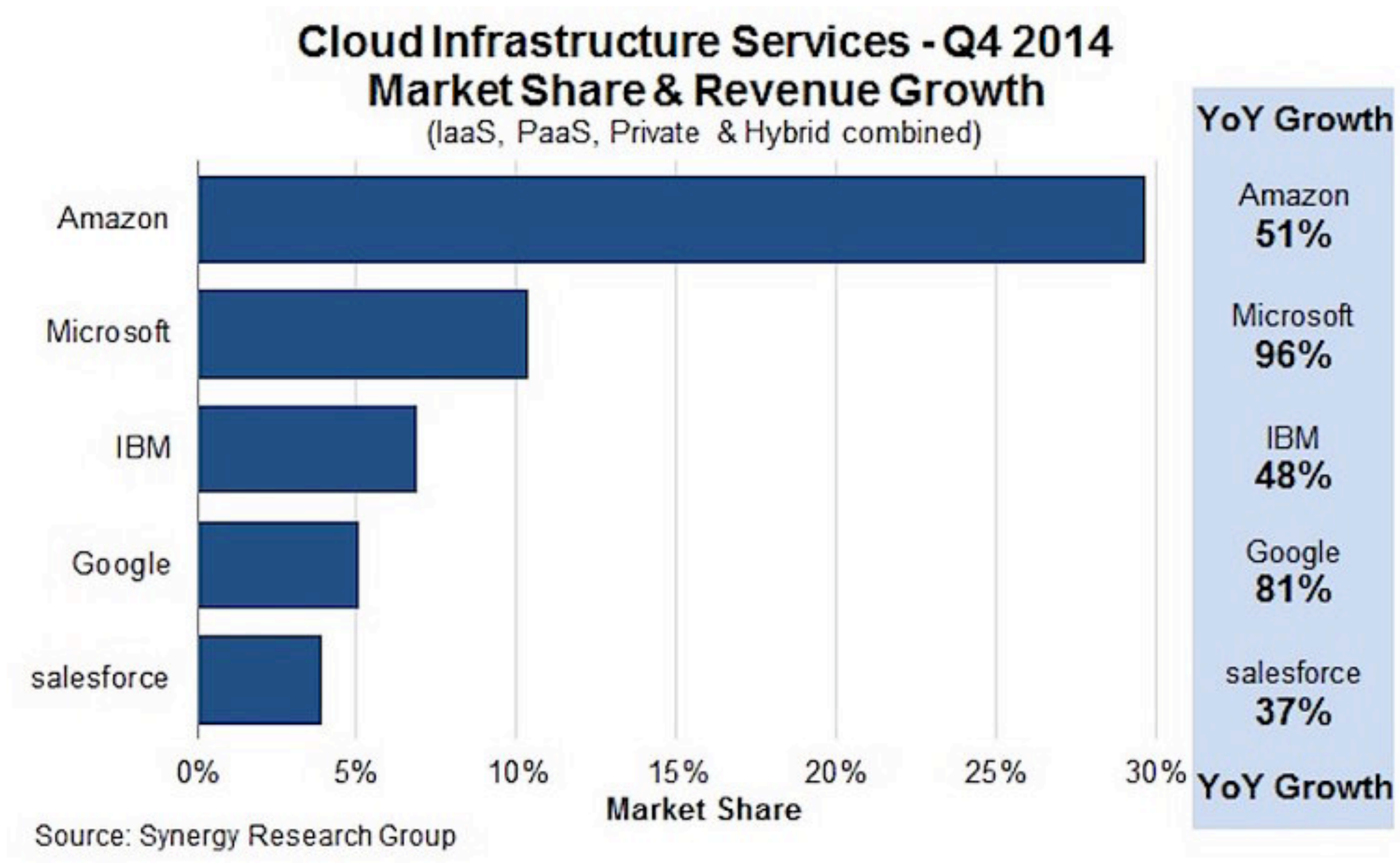
  - *We will be using all of these*

Cloud Computing

- So basically- the rest is kind of history. The growth of AWS is, well:

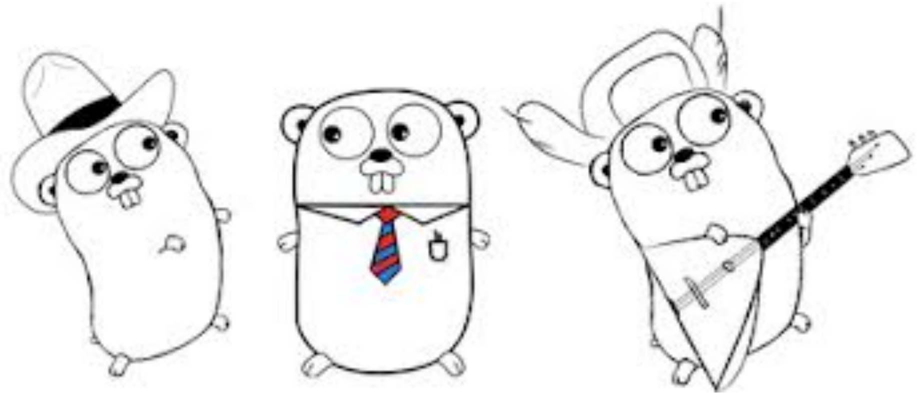


AWS history (continued)



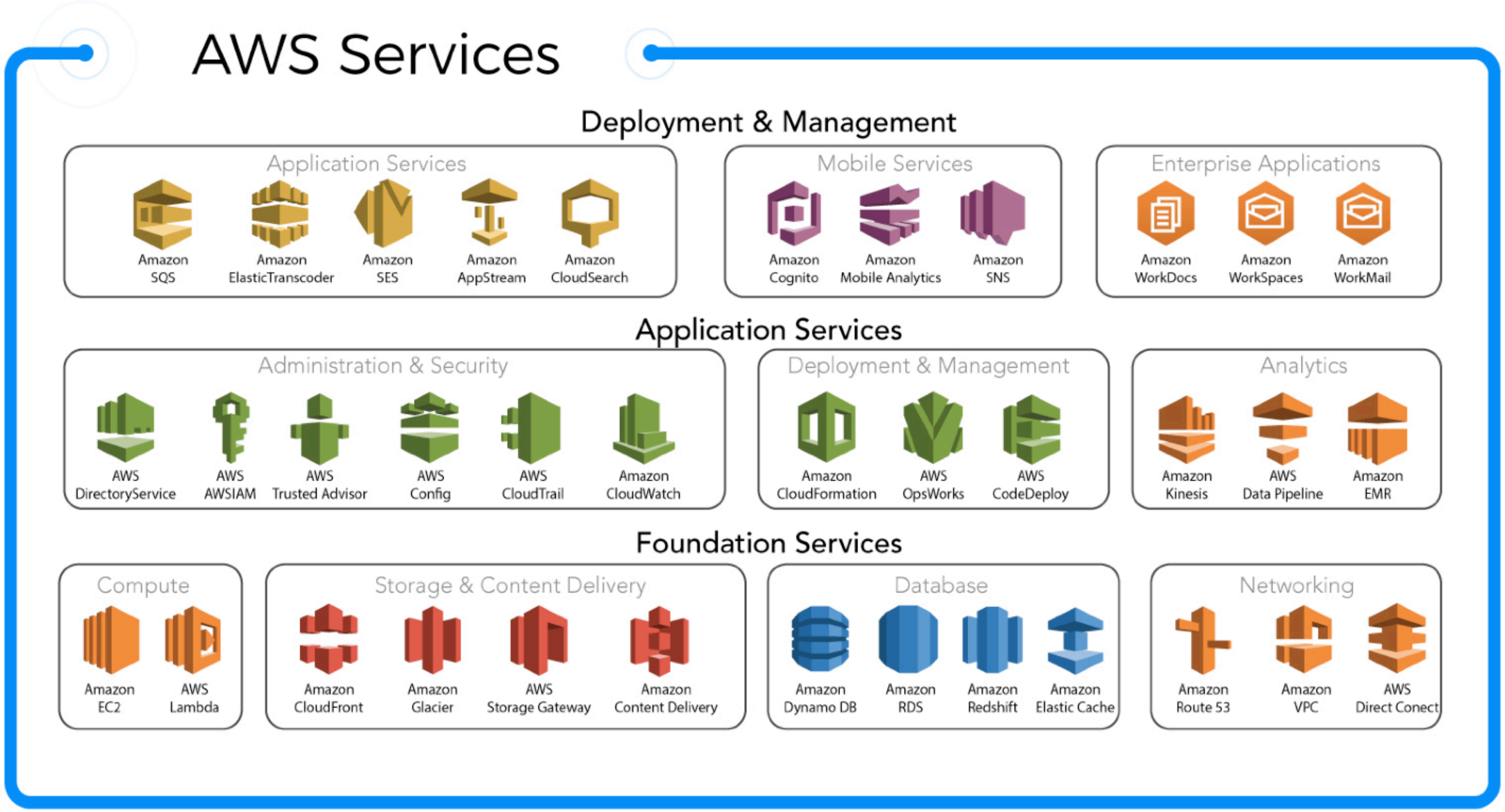
AWS Philosophy: Serverless

- Obviously AWS continued to add services consistently in the years since. This has led to some some interesting consequences and coding philosophies. We’re going to go over some of those now as they will be extremely relevant in our upcoming modules.



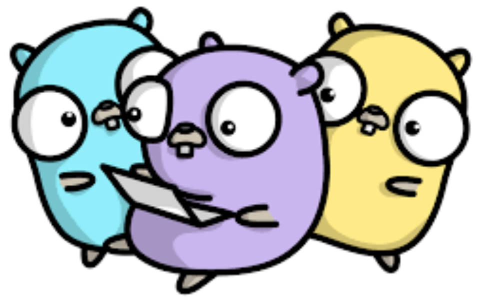
AWS Services (categorized)

Develop Intelligence



Serverless Philosophy

- So as a consequence of all of these various services being created by AWS we have seen a plethora of applications adopting a *serverless* or *microservices* philosophy.
- In layman's terms this basically means that instead of depending on a monolithic server to run your application you break it down to smaller portions and let AWS manage those portions.



Microservices Advantages

- So what is the advantage of utilizing a microservices architecture?
  - The follow the Single Responsibility principle
  - Resilient/Flexible (you can have a failure in one service without crashing the others)
  - Highly scalable
  - Easy to enhance (as you are dealing with less dependencies)
  - Low impact on other services
  - Easier to understand
  - Easier to Test each unit individually
  - Easier to deploy
  - Free to choose technology

Microservices Disadvantages

- But there can be disadvantages, including:
  - Tougher to manage multiple deployments
  - Integrations between disparate services can have unexpected consequences
  - Integration testing can be painful
  - It quickly becomes an administrative nightmare managing the deployments of 10,000 pieces of separate code instead of one monolithic one.

Microservices as modern architecture

- Overall microservices are the direction that the industry seems to be taking. Everything from DOCKER (self contained containers intended to wrap around a piece of code and do A thing) to utilizing AWS managed services point to this as the future of coding.
- This should hopefully help with reducing the 3am calls to dev/ops when the servers go down.

Another consequence….

- Another consequence of the modern ”serverless” architecture is increased complexity.
- In some ways it’s nice not to have to worry about a single server…but what you’ve traded that off for (in the dev/ops world) is that you now have to worry about 50 servers…in the form of microservices.
- How can we make it easier for the dev/ops team to manage multiple servers?

The AWS Well Architected Framework

- So the AWS well architected framework is designed to help you understand the pros and cons of decisions that you make while building systems on AWS.
- The idea here is that we create a framework that forces a consistent approach to development decisions made by architects, solutions architects, CEOs, and business managers.

The Pillars of the W.A.F

Pillar Name	Description	AWS Tools
Operational Excellence	The ability to run and monitor systems to deliver business value and to continually improve supporting processes and procedures.	AWS Config, AWS X-ray
Security	The ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies.	AWS VPC, Security Groups
Reliability	The ability of a system to recover from infrastructure or service disruptions, dynamically acquire computing resources to meet demand, and mitigate disruptions such as misconfigurations or transient network issues.	Cloudformation, Terraform
Performance Efficiency	The ability to use computing resources efficiently to meet system requirements, and to maintain that efficiency as demand changes and technologies evolve.	AWS Lambda, AWS Fargate, DynamoDB, S3
Cost Optimization	The ability to run systems to deliver business value at the lowest price point.	AWS Cloudtrail

Terms

Term	Definition
Component	The code, configuration and AWS Resources that together deliver against a requirement. A component is often the unit of technical ownership, and is decoupled from other components
Workload	A set of components that together deliver business value. The workload is usually the level of detail that business and technology leaders communicate about.
Milestones	These mark key changes in your architecture as it evolves throughout the product lifecycle (design, testing, go live, and in production).
Architecture	How components work together in a workload. How components communicate and interact is often the focus of architecture diagrams.
Technology portfolio	The collection of workloads that are required for the business to operate.

General Design Principles

- Stop guessing your capacity needs
  - Flexibility is key. Guessing leads to idle time or scrambling and the associated high costs
- Test systems at production scale
  - This is an advantage of keeping your infrastructure-as-code: you can set up test infrastructure and test it at scale then tear it down to save money
- Automate to make architectural experimentation easier
  - Automate deployments. Automate Integrations. Automate tests.
- Allow for evolutionary architectures
  - This means that you should have different components decoupled enough to swap out, for example, a Postgres DB for DynamoDB tables feeding a Kinesis stream
- Drive architectures using data
  - Use AWS tools to measure performance with things like Cloudwatch. Look at speeds, invocations and costs to decide where to make improvements.
- Improve through game days
  - Schedule regular tests with wonky data to see how the system reacts. Simulate production.

Pillar One: Operational Excellence

- This one is about the ability to run and monitor systems to deliver business value whilst continually improving supporting processes and procedures.
- Continuous Improvement is a big part of this- it means that you are constantly evaluating your systems and looking for opportunities to improve the code, the performance, or the architecture.

Six Principles of Operational Excellence

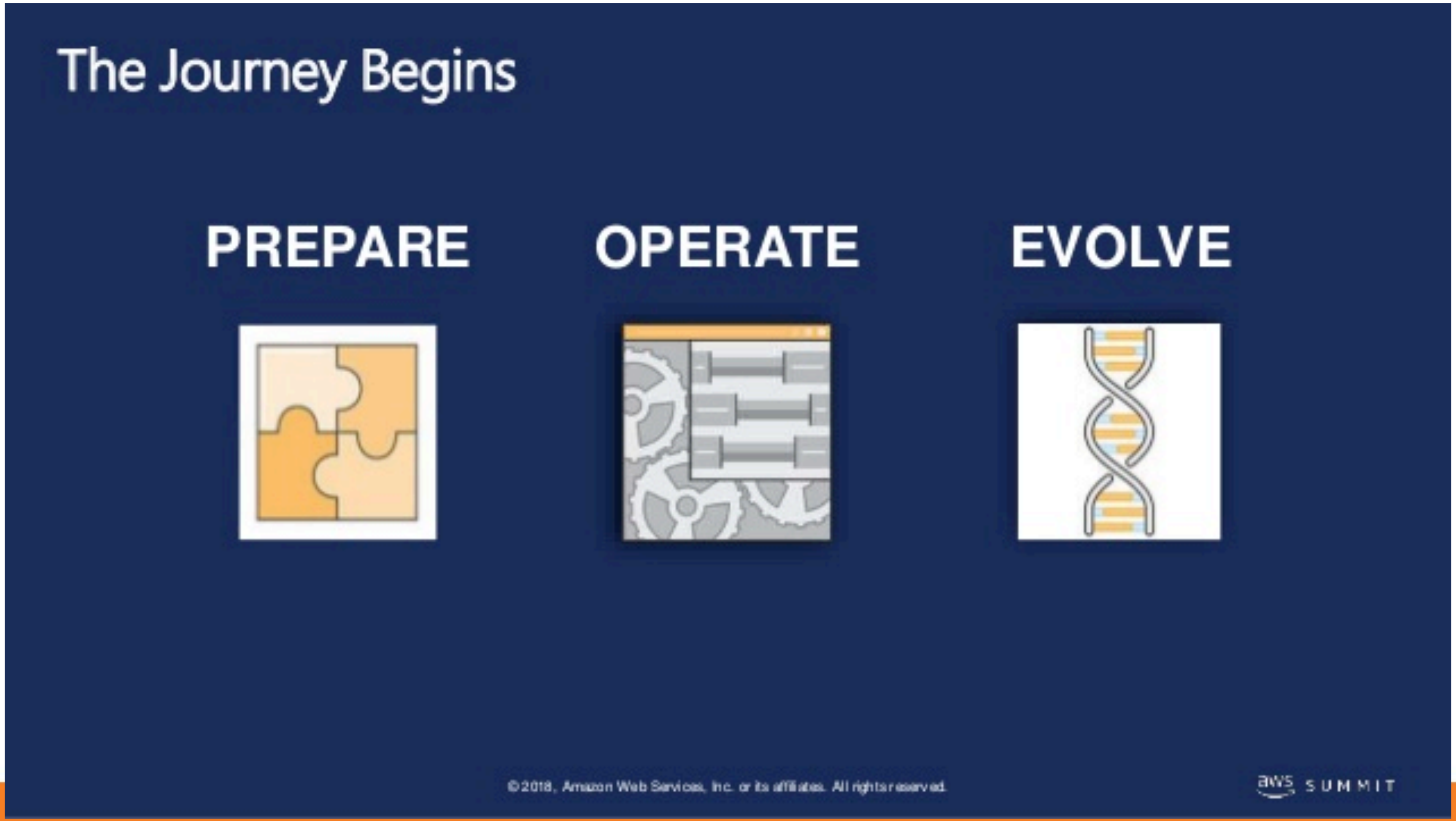
Principle	Description
Perform operations as code	In the cloud, you can apply the same engineering discipline that you use for application code to your entire environment. You can define your entire workload (applications, infrastructure) as code and update it with code. You can implement your operations procedures as code and automate their execution by triggering them in response to events. By performing operations as code, you limit human error and enable consistent responses to events.
Annotate documentation	In an on-premises environment, documentation is created by hand, used by people, and hard to keep in sync with the pace of change. In the cloud, you can automate the creation of annotated documentation after every build (or automatically annotate hand-crafted documentation). Annotated documentation can be used by people and systems. Use annotations as an input to your operations code.

Six Principles of Operational Excellence

Principle	Description
Refine operations procedures frequently	As you use operations procedures, look for opportunities to improve them. As you evolve your workload, evolve your procedures appropriately. Set up regular game days to review and validate that all procedures are effective and that teams are familiar with them
Make frequent, small, reversible changes	Design workloads to allow components to be updated regularly. Make changes in small increments that can be reversed if they fail (without affecting customers when possible)
Anticipate failure	Perform “pre-mortem” exercises to identify potential sources of failure so that they can be removed or mitigated. Test your failure scenarios and validate your understanding of their impact. Test your response procedures to ensure that they are effective, and that teams are familiar with their execution. Set up regular game days to test workloads and team responses to simulated events
Learn from all operational failures	Drive improvement through lessons learned from all operational events and failures. Share what is learned across teams and through the entire organization.

Process

- Nothing especially profound about the process here; Amazon is a big fan of the Prepare, Operate, Evolve


The Journey Begins
PREPARE
OPERATE
EVOLVE
©2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
aws SUMMIT

Practical applications of POE

- PREPARE: Use AWS Config- especially the AWS Config RULES to make sure that everything going on with your infrastructure is as intended. We'll do a quick demo here for that…
- OPERATE: We'll use AWS Cloudwatch for this- which we will get to shortly. Also AWS X-Ray
- EVOLVE: AWS Elasticsearch Service is awesome for this type of stuff.

Pillar Two: Security

- So for the Security principal we'll want to think about protecting information, systems, and assets. We'll do this by following several design principals (just like with Operational Excellence).

Security Design Principals

Design Principle	Definition	AWS Tools
Implement a strong identity foundation	Implement the principle of least privilege and enforce separation of duties with appropriate authorization for each interaction with your AWS resources. Centralize privilege management and reduce or even eliminate reliance on long-term credentials.	IAM Identity Management
Enable traceability	Monitor, alert, and audit actions and changes to your environment in real time. Integrate logs and metrics with systems to automatically respond and take action.	IAM
Apply security at all layers	Rather than just focusing on protection of a single outer layer, apply a defense-in-depth approach with other security controls. Apply to all layers (e.g., edge network, VPC, subnet, load balancer, every instance, operating system, and application).	VPC, subnets, security groups, load balancers
Automate security best practices	Automated software-based security mechanisms improve your ability to securely scale more rapidly and cost effectively. Create secure architectures, including the implementation of controls that are defined and managed as code in version-controlled templates.	Terraform, Cloudformation
Protect data in transit and at rest	Classify your data into sensitivity levels and use mechanisms, such as encryption, tokenization, and access control where appropriate.	SSM, API Gateway, Cloudformation, Cognito

Security Design Principles

Design Principle	Definition	AWS Tools
Keep people away from data	Create mechanisms and tools to reduce or eliminate the need for direct access or manual processing of data. This reduces the risk of loss or modification and human error when handling sensitive data.	RDS, S3, Azure, Redshift, AWS Encryption keys, KMS service
Prepare for security events	Prepare for an incident by having an incident management process that aligns to your organizational requirements. Run incident response simulations and use tools with automation to increase your speed for detection, investigation, and recovery	Lambdas, EC2, AWS Guard-Duty

Security Design Principles

- We'll be going over some of the tools we'll be using in security design in the upcoming sections (around modules four and two)…but let's list some of them and do a quick intro here:
  - AWS Web Application Firewall: AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to an Amazon API Gateway API, Amazon CloudFront or an Application Load Balancer. AWS WAF also lets you control access to your content
  - AWS Sheild: Protection against DDOS attacks- comes standard with AWS resources.
  - AWS Macie: AWS machine learning that automatically detects, classifies, and protects sensitive data using machine learning.
  - AWS Key Management System: Encrypt data

Reliability Pillar

- So for the Reliability pillar we want the system to recover from infrastructure or service disruptions, dynamically require new resources as necessary to meet demand and mitigate disruptions due to network issue.

Reliability Design Principles

Principle	Description	AWS Tools
Test recovery procedures	In an on-premises environment, testing is often conducted to prove the system works in a particular scenario. Testing is not typically used to validate recovery strategies. In the cloud, you can test how your system fails, and you can validate your recovery procedures. You can use automation to simulate different failures or to recreate scenarios that led to failures before. This exposes failure pathways that you can test and rectify before a real failure scenario, reducing the risk of components failing that have not been tested before.	Cloudformation and Terraform
Automatically recover from failure	By monitoring a system for key performance indicators (KPIs), you can trigger automation when a threshold is breached. This allows for automatic notification and tracking of failures, and for automated recovery processes that work around or repair the failure. With more sophisticated automation, it's possible to anticipate and remediate failures before they occur.	Cloudwatch Alarms, Cloudtrail, X-Ray, Cloudline, KMS
Scale horizontally to increase aggregate system availability	Replace one large resource with multiple small resources to reduce the impact of a single failure on the overall system. Distribute requests across multiple, smaller resources to ensure that they don't share a common point of failure.	Microservices so Lambda, Fargate, Docker, s3

Reliability Resign Principles

Principle	Description	AWS Tools
Stop guessing capacity	A common cause of failure in on-premises systems is resource saturation, when the demands placed on a system exceed the capacity of that system (this is often the objective of denial of service attacks). In the cloud, you can monitor demand and system utilization, and automate the addition or removal of resources to maintain the optimal level to satisfy demand without overor under- provisioning.	Dynamo, RDS, Aurora, Lambda, Fargate
Manage change in automation	Changes to your infrastructure should be done using automation. The changes that need to be managed are changes to the automation	Terraform, Serverless

Performance Efficiency

- The performance efficiency pillar refers to the use of aws resources efficiently to meet system requirements while maintaining efficiency as demand changes and technology evolves.
- This basically means- don’t create too much or too little- where possible keep it elastic.

Performance Efficiency

Principle	Description	AWS Tools
Democratize advanced technologies	Technologies that are difficult to implement can become easier to consume by pushing that knowledge and complexity into the cloud vendor's domain. Rather than having your IT team learn how to host and run a new technology, they can simply consume it as a service. For example, NoSQL databases, media transcoding, and machine learning are all technologies that require expertise that is not evenly dispersed across the technical community. In the cloud, these technologies become services that your team can consume while focusing on product development rather than resource provisioning and management	Dynamo, S3, RDS
Go global in minutes	Easily deploy your system in multiple Regions around the world with just a few clicks. This allows you to provide lower latency and a better experience for your customers at minimal cost	AWS CI/CD Pipeline
Use serverless architectures	In the cloud, serverless architectures remove the need for you to run and maintain servers to carry out traditional compute activities. For example, storage services can act as static websites, removing the need for web servers, and event services can host your code for you. This not only removes the operational burden of managing these servers, but also can lower transactional costs because these managed services operate at cloud scale.	Lambda, Fargate, RDS, Elasticache

Performance Efficiency

Principle	Description	AWS Tools
Experiment more often	With virtual and automatable resources, you can quickly carry out comparative testing using different types of instances, storage, or configurations.	ALL
Mechanical sympathy	Use the technology approach that aligns best to what you are trying to achieve. For example, consider data access patterns when selecting database or storage approaches	Cloudwatch, VPC

Performance Efficiency

- Compute: Auto Scaling is key to ensuring that you have enough instances to meet demand and maintain responsiveness.
- Storage: Amazon EBS provides a wide range of storage options (such as SSD and provisioned input/output operations per second (PIOPS)) that allow you to optimize for your use case. Amazon S3 provides serverless content delivery, and Amazon S3 transfer acceleration enables fast, easy, and secure transfers of files over long distances.
- Database: Amazon RDS provides a wide range of database features (such as PIOPS and read replicas) that allow you to optimize for your use case. Amazon DynamoDB provides single-digit millisecond latency at any scale.

Performance Efficiency

- Network: Amazon Route 53 provides latency-based routing. Amazon VPC endpoints and AWS Direct Connect can reduce network distance or jitter.
- Monitoring: Amazon CloudWatch provides metrics, alarms, and notifications that you can integrate with your existing monitoring solution, and that you can use with AWS Lambda to trigger actions.
- Tradeoffs: Amazon ElastiCache, Amazon CloudFront, and AWS Snowball are services that allow you to improve performance. Read replicas in Amazon RDS can allow you to scale read-heavy workloads.

Cost Optimization

- The cost optimization pillar is about running your systems efficiently at the lowest possible price point.

Cost Optimization

Principle	Description
Adopt a consumption model	Pay only for the computing resources that you require and increase or decrease usage depending on business requirements, not by using elaborate forecasting. For example, development and test environments are typically only used for eight hours a day during the work week. You can stop these resources when they are not in use for a potential cost savings of 75% (40 hours versus 168 hours).
Measure overall efficiency	Measure the business output of the workload and the costs associated with delivering it. Use this measure to know the gains you make from increasing output and reducing costs.
Stop spending money on data center operations	AWS does the heavy lifting of racking, stacking, and powering servers, so you can focus on your customers and organization projects rather than on IT infrastructure.
Analyze and attribute expenditure	The cloud makes it easier to accurately identify the usage and cost of systems, which then allows transparent attribution of IT costs to individual workload owners. This helps measure return on investment (ROI) and gives workload owners an opportunity to optimize their resources and reduce costs.
Use managed and application level services to reduce cost of ownership	In the cloud, managed and application level services remove the operational burden of maintaining servers for tasks such as sending email or managing databases. As managed services operate at cloud scale, they can offer a lower cost per transaction or service.