IST346
Cloud Computing

What is This “Cloud” thing?

Cloud Computing

- “The practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer.”

- In other words, having someone else host your servers and/or services for you.

Cloud Services Overview

- Typically utilizes Virtualization or Containers to keep costs down while offering a larger number of services.
- Shared resource model helps make the best use of hardware.
- Providers have varying chargeback models to customers:
  - By amount of data stored (per mb/gb/tb)
  - By bandwidth used (mb/hr, gb/week, gb/month, etc..)
  - By resources used (ram, cpu)
  - By number of users accessing the service
  - Or combinations of all of the above.

Benefits of Cloud Services

- **Costs** - Helps companies avoid up front costs of infrastructure
- **Anywhere/anytime** - You can use the service whenever and wherever you wish (provided you have an internet connection)
- **Management** - Servers/services can be partially or fully managed by the cloud provider
- **Network** - Cloud computing can deliver increased bandwidth
- **Economies of Scale** - Cloud providers take advantage of the sharing of redundant resources to reduce costs and increase scale.
- **Scalability** - In addition it is easy to scale cloud services because providers have already invested in infrastructure.
- **Deployment** - quick deployment compared to in house solutions.
- **Backup** – Backup and recovery are sped up and simplified since the data resides at the cloud provider.

Drawbacks to Cloud Services

- **Big data** - Not easy to move huge datasets in and out of the cloud.
- **Control** - Reduced control for customers who want or need it.
- **Integration** - The silo effect exists; when you have multiple cloud services hosting your data, integration across them and with your own network can be daunting (if possible at all).
- **Security** – how do you know your data is safe?
- **Costs** - Increased long term costs over in house solutions.
- **Customization** - Can reduce flexibility of services offered
- **Support** - Places reliance on cloud provider to fix issues
- **Features** – often features are not on par with in house solutions
- **Regulatory Compliance** – some industries are not permitted to store data off premises.

Cloud Service Models

Cloud Services on the Internet

- **Cloud computing** is an extension of the service model to the ubiquities of the internet.

- Don’t want to deal with datacenters or servers? Try **Infrastructure as a Service!**
- Don’t want to bother with the infrastructure and the components required by your service? Try **Platform as a Service!**
- Heck, don’t want to bother with any of it? Then **Software as a Service** is for you!

Infrastructure as a Service (IAAS)

- The most basic cloud-service model
- Provides you with virtual infrastructure, for example servers and data storage space.
- Virtualization plays a major role in this mode, by allowing IaaS-cloud providers to supply resources on-demand extracting them from their large pools installed in data centers.
- High level of flexibility, but requires greater IT skillsets.
- **Sales Pitch:** “Let us handle your hardware (as virtual machines). You handle the rest”
- **Examples:** Amazon EC2, GoGrid, RackSpace

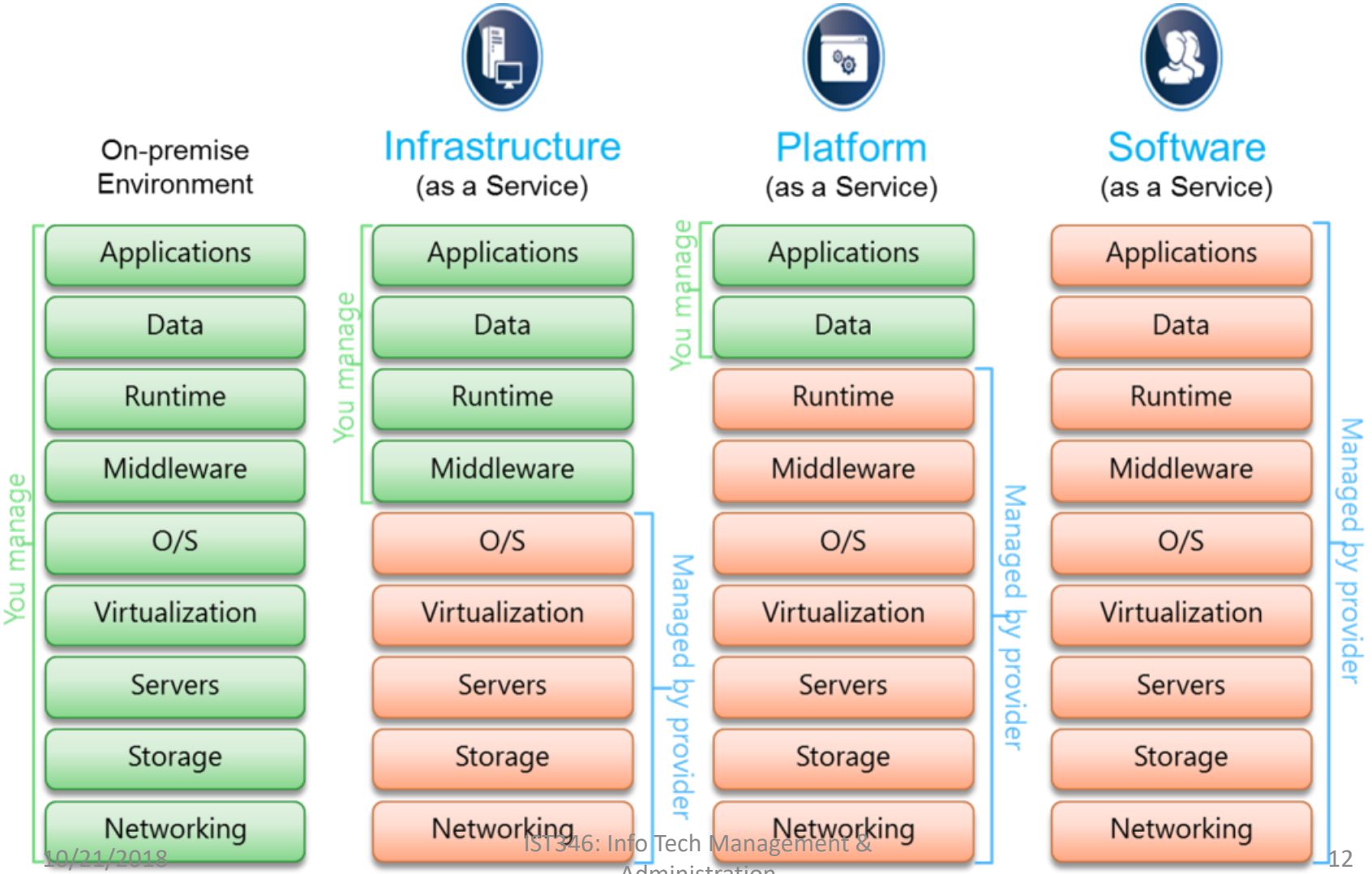
Platform as a Service (PAAS)

- Cloud providers deliver you the development environment for services where the user can develop and run in-house built applications.
- These services might include an operating system, a programming language execution environment, databases and web servers.
- Still offers shops flexibility to develop and customize their applications, but requires less of a skillset than IAAS as they no longer need to know how to setup and manage the infrastructure and Operating Systems.
- Containers in the cloud are an example of PaaS
- **Sales Pitch:** "We give you a setup so you can install or build your own app. No need to worry about the system administration."
- **Examples:** Google App Engine, Heroku, Microsoft Azure App Service, Docker Cloud

Software as a Service (SAAS)

- Provides you with access to already developed applications that are running in the cloud.
- The access is achieved by cloud clients and the cloud users do not manage the infrastructure where the application resides, eliminating with this the way the need to install and run the application on the cloud user's own computers.
- This is method requires the least amount of IT skillsets in house, but also reduces or eliminates flexibility and control over how the service should function.
- **Sales Pitch:** "We give you the apps, all you need to do is your job!"
- **Examples:** Salesforce.com, Office 365, Google Apps, Quickbooks Pro online, Draw.io

Cloud Service Models Comparison



Other Models

- **FaaS** – Functions as a Service / Serverless computing.
  - Run your code in the cloud without the worry of managing IaaS or even PaaS / Containers
  - Ideal for microservices architecures
  - **Examples:** AWS Lambda, OpenFaaS, Google Cloud Functions
- **DaaS** – Desktop as a Service
  - Cloud-based VDI (Virtual desktop infrastructure)
  - **Examples:** Mac-In-Cloud, Windows Virtual Desktop, VMWare Workspaces

The Cloud and D.R. (Disaster Recovery)

DR sites in the Cloud

The Claim:

“Cloud Computing delivers faster recovery times and multi-site availability at a fraction of the cost of conventional disaster recovery.”

- Can provide bare-metal recovery that includes all services and data
- Ability for more rapid recovery than typical failover sites
- Potential for automated failover
- Allows for backing up of data off site
- Increased mobility, you can connect to “the cloud” from anywhere.

Hot Sites

- Hot site - commercial disaster recovery service that allows a business to continue computer and network operations in the event of a computer or equipment disaster.
- Example: If an enterprise's data center becomes inoperable, they can quickly switch all data processing operations to a hot site with little to no downtime.
- Provides complete duplication of primary environment and will be available at all times.
- More expensive, but faster to recover a business in the event of a disaster.

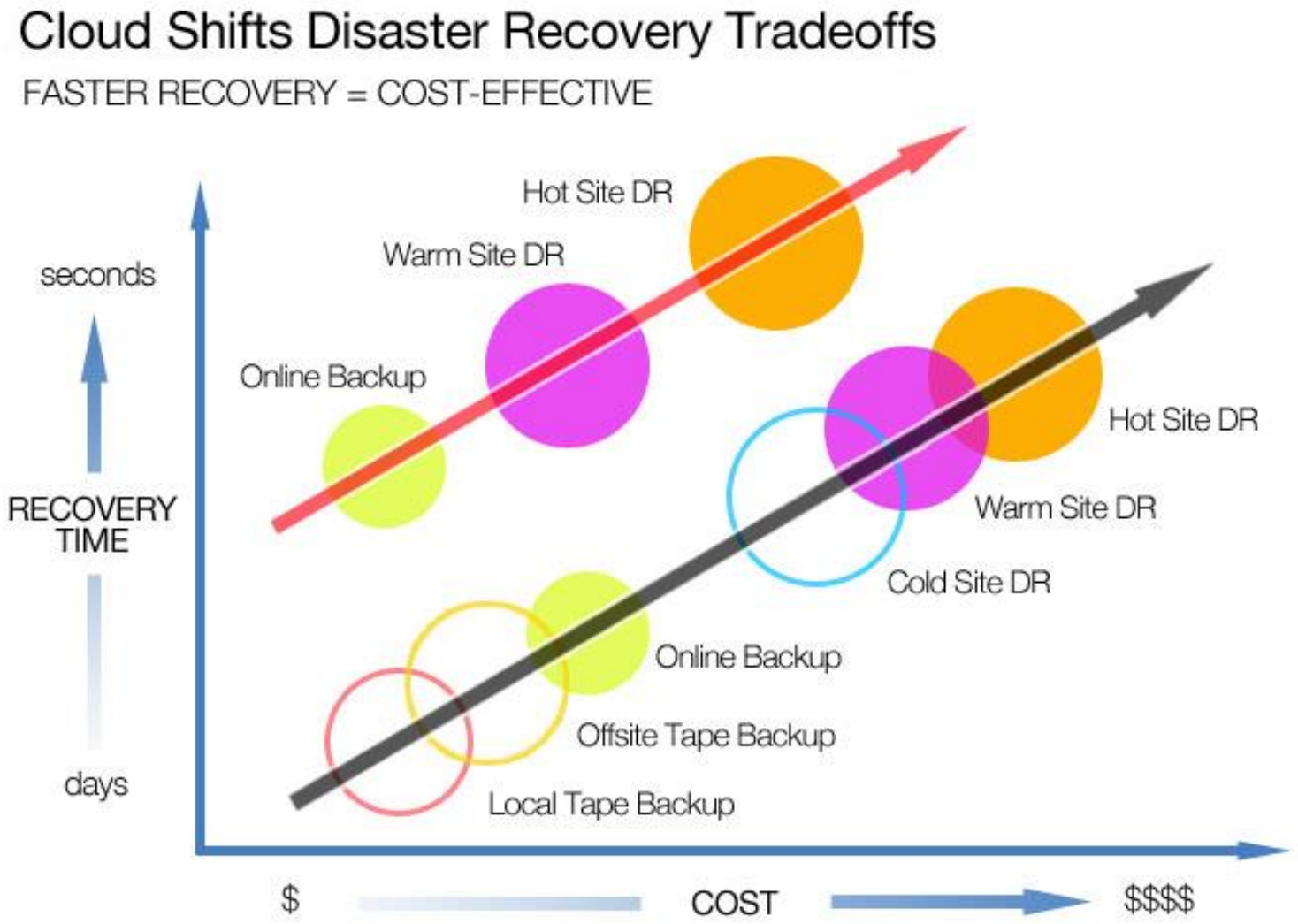
Warm Sites

- A warm site is backup site that is setup similar to a hot site, but requires work to render it functional in the event of a disaster.
- Backup environment are not constantly operational or processing. Also may require networking changes to redirect server/service requests to the backup site.
- Requires intervention and cannot accomplish seamless failover, but cheaper to employ than a hot site.
- While they don’t offer immediate failover, warm sites are typically easier to maintain than hot sites due to lack need for real-time replication or complex networking schemes.

Cold Sites

- A cold site is a type of disaster recovery service that provides backup, but the site isn't configured to take over in the event of a disaster.
- Downtimes are to be expected and may require considerable amount of work for bringing a site online.
- A cold site is less expensive, but it takes longer to get an enterprise in full operation after the disaster.

Costs vs. Recovery Time



SLAs and the Cloud

Service Level Agreements - Cloud

- SLAs set expectations for both parties (client and cloud provider)
- Identifies the specific parameters and minimum levels required for each element of a service.
- Affirms your ownership of data stored on a cloud provider's system, and your rights to get it back.
- Details the infrastructure and security standards to be maintained.
- Specifies your rights and costs to continue and discontinue use of the service.

What do we need to be concerned with?

- **Availability** (e.g. 99.99% during work days, 99.9% for nights/weekends)
- **Performance** (e.g. maximum response times)
- **Security / privacy** of the data (e.g. encrypting all stored and transmitted data)
- **Disaster Recovery** expectations (e.g. worse case recovery commitment)
- **Location of the data** (e.g. consistent with local legislation)
- **Access to the data** (e.g. data retrievable from provider in readable format)
- **Portability of the data** (e.g. ability to move data to a different provider)
- **Support and Problem Resolution** (e.g. call center)
- **Change Management** process (e.g. changes – updates or new services)
- **Dispute mediation** process (e.g. escalation process, consequences)
- **Exit Strategy** with expectations on the provider to ensure smooth transition

Lessons to be learned with SLAs

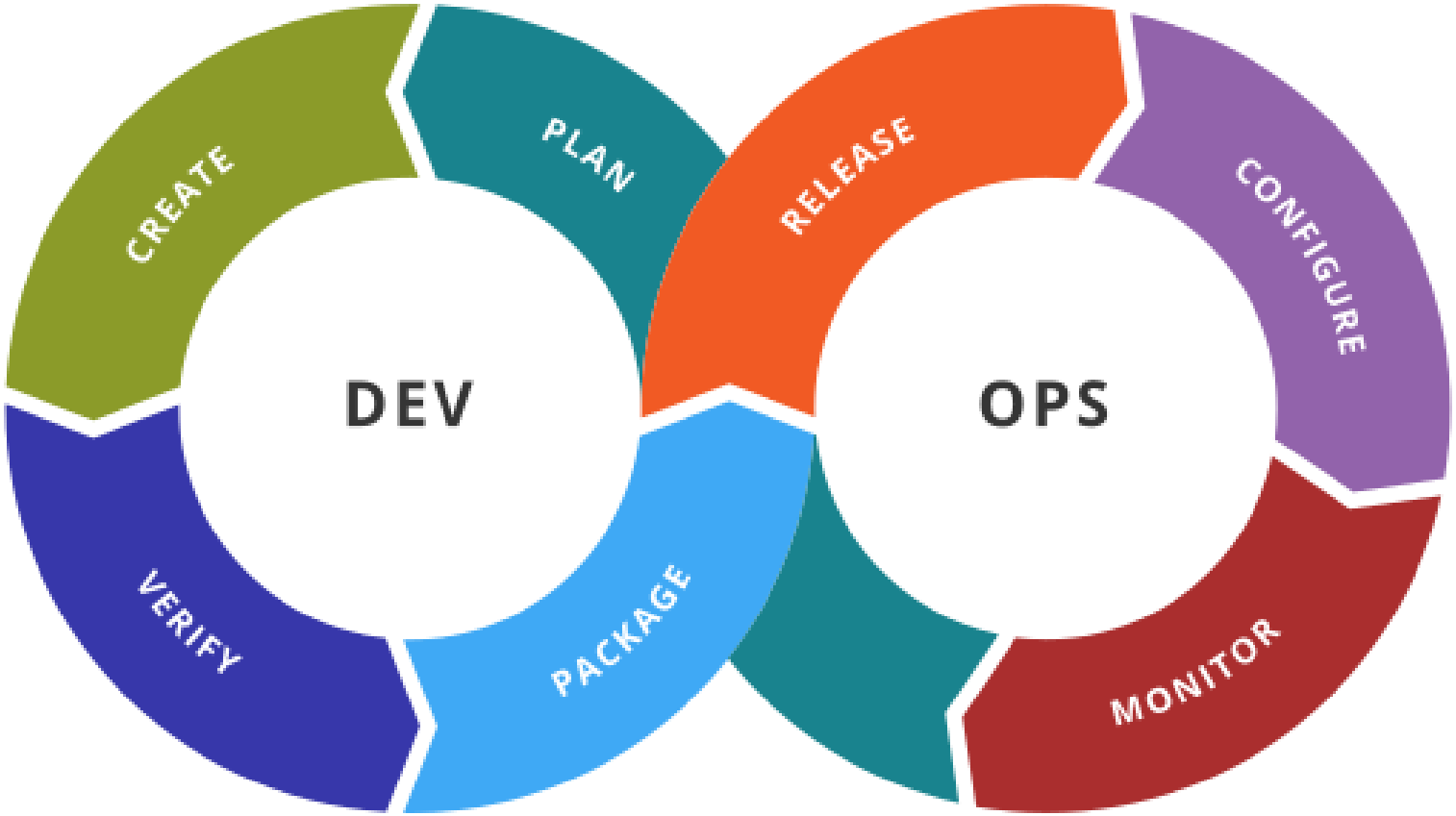
- Read cloud provider's SLA very carefully.
  - Devil's in the details, if you don't read it you won't know that they aren't getting your data back right away until the event of a failure, in which case it's too late.
- Get technical staff involved to validate SLAs against common outage scenarios.
  - Just like IT staff should have outage and DR plans for on-premise services, they should help develop the same for your cloud vendors.
- Have contingency plans in place for worst case scenarios.
  - Have a business continuity plan in place for when something really bad happens to your cloud vendor's service. Blindly putting the trust of your entire business in a vendor's hands is asking for trouble.

DevOps

What is DevOps?

- A set of practices to reduce the time between making a change to a system and realizing that change in production (whether on-premises or in the cloud) without sacrificing quality of system stability.
- Historically slowed down by:
  - Transitioning to Dev / Test / Prod environments
  - Pet Servers
  - Organizational Culture / Mindset
  - Siloed teams

The DevOps Cycle



https://en.wikipedia.org/wiki/DevOps#/media/File:Devops-toolchain.svg

Core Values of DevOps - CAMS

- **Culture** – breaking down barriers between teams, shortening feedback loops
- **Automation** – productivity gains in deployment, systems thinking
- **Measurement** – basing decisions on data instead of guessing
- **Sharing** – tooling, discoveries and lessons among the team

DevOps Goals

- **Systems thinking** - imagine your systems as a whole not as parts
- **Culture of ownership** – everyone involved has ownership over the entire system and process
- **Shortening feedback loops** - take less time to fix problems and achieve goals.
- **Culture of experimentation and learning** - no fear of making changes as you should be able to test them easily.

Infrastructure as Code Methodology for DevOps

- A
- Treat your infrastructure as if it were code!
- Store configurations, dependencies and scripts to bootstrap your systems in a source code management (SCM) system like Git.
- This allows you set-up and tear down environments quickly and easily and deploy your systems in Dev, Test or Production.
- **Servers are commodity / utility resources, and not at all strategic. A perfect scenario for cloud computing!**

Recall: Treat Servers like Cattle, Not Pets

- Pet servers
  - Few of them
  - Are given names like mail.mycompany.com
  - Are built to solve the task at hand (email)
  - When they are “sick” we “nurse them back to health”
- Cattle servers:
  - Lots of them
  - Are given numbers like s0045.mycompany.com
  - Are built to do the same thing: compute and storage
  - When they are “sick” we “take them out of commission” and replace with another.
  - Cloud Computing Mindset

Essential DevOps Toolbox for IaC

- **Source Code Management** (git, subversion, mercurial)
- **Virtualization / Containerization** (VMWare, Vagrant, Docker, CoreOS)
- **Configuration Management** (Ansible, Chef, Puppet, Docker-Compose)
- **Orchestration** (Kubernetes, Mesos, Rancher, Docker Swarm)
- **Continuous Integration / Continuous Delivery** (Jenkins, TeamCity)
- **Monitoring / Logging** (Monit, Nagios, ELK Stack)