Cloud vs. On Premise

Health Tech Net

17 June 2022

James (jc) Conley, AWS Sr. TAM
jameconl@amazon.com

What is the ‘Cloud’ To You?

- Phone Storage
- Location of your Web Site
- Out Sourced Computing or Applications
- Remote Access to you Information Technology (IT)

“My same applications, just remote and out sourced (i.e. not my employees)”

The New Paradigm - Breakdown

- Cost, Cost, Cost
  - *Most people focus here – we will talk about this last*
- Business Approach – Application Development
  - *Probably where you will realize the greatest gain / change*
- Operate (Steady State)
  - *The hardest area for new customers to adopt / accept*
- Reliability (When the Un-expected Happens)
  - *Brings capability that (realistically) is only achievable in the Cloud*

Other pillars not addressed: *Security, Sustainability, Governance/Compliance*

Business Approach

Normal:

- Provide executive sponsorship
- Evaluate threat landscape
- Identify owners (resources, processes)
- Resource teams appropriately
- Predefine responsibilities between teams
- Encourage escalation

- Establish mechanism to :
  - identify responsibility and ownership
  - request additions, changes, and exceptions
- Ensure team members are:
  - Empowered to act when outcomes are at risk
  - Enabled and encouraged to grow their skill sets

Cloud: Take advantage of the low cost for experimenting and validating

- Encourage experimentation; evaluate tradeoffs
- Identify owners responsible for operational activities
- Ensure team members know their responsibilities
- Pay-as-your-Go: strive for 0% overhead and 100% return on investment

Development Principles

Normal DevOps:

- Implement telemetry (application, workload, user activity, dependency, transaction)
- Design should be data driven – Key Performance Indicators (KPI)
- Implement practices to improve code quality
- Ensure consistent review of operational readiness
- Implement Continuous Improvement, Continuous Development (CI/CD)

Cloud: Take advantage of the low cost of environments

- Implement * as code
- Experiment using limited deployments – *Try before you Buy*
- Test and validate changes – at scale
- Deploy using parallel environments
- Design-in elasticity – start at Min usage – scale as needed
- Design-in High Availability (HA) and Disaster Recovery (DR) – *at no cost*

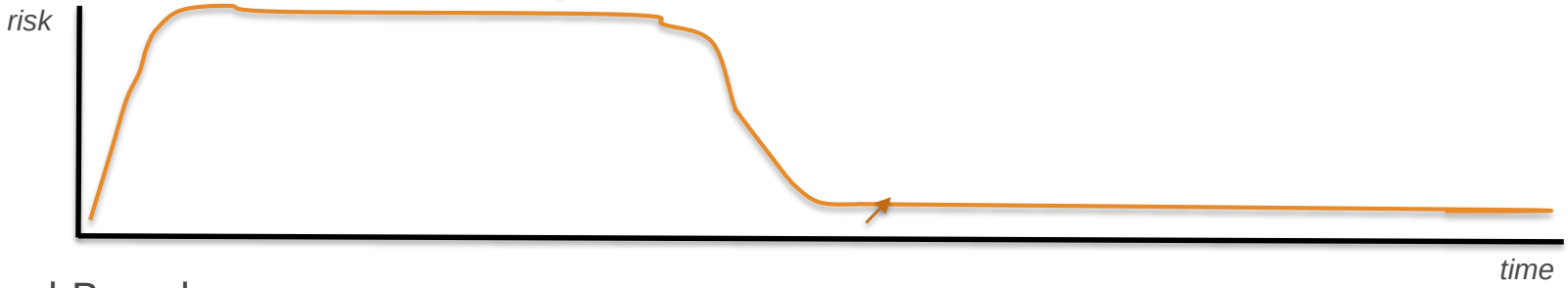
Development – PROs

- Experiment, Experiment, Experiment
- Design in a sandbox, not on paper; Use labor hours to code not debate
- Use of cloud services supports a better micro-service design/adoption
- Easy adoption of managed services that might otherwise not be considered
- Take advantage of *Serverless* capabilities (only available in the cloud)
- Develop Cattle not Pets
- Architect for Disaster Recovery – Implement * as code
- Lower risk with flexible development (next slide)

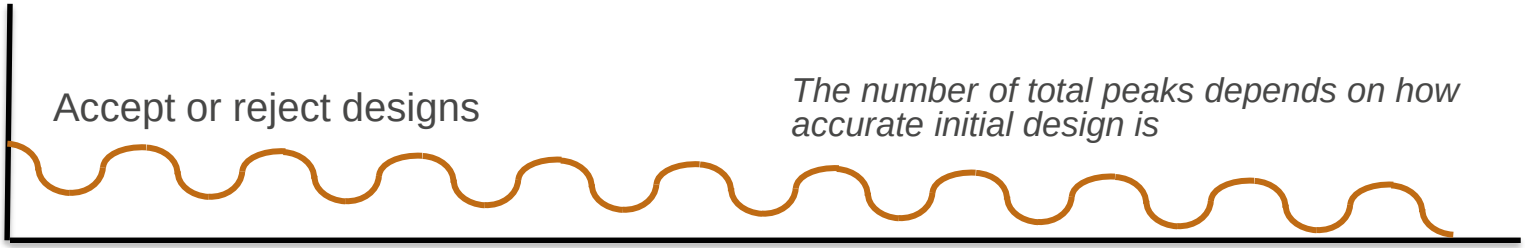
Development – Total Risk Accounting

On-Premise

Length of plateau depends on how accurate initial design is



Cloud-Based



Development – CONs

- Personnel – A lot to learn; skills can be acquired, but it takes time
  - 3 months min; 1 year for adequate
- You may want to change your development ‘Partner’ (contractor)
  - Smaller, agile, and more cloud-savvy is best
- You will, eventually, want to re-work your legacy systems/applications
- Cloud-specific design considerations -> changes result in time and $$s
  - More micro-services than you might do on-premise
  - Design for Hybrid or Cloud Agnostic
  - Cattle not Pets can be very difficult
- Test weak points early and at full scale
  - Work with Cloud experts to understand what is a cloud weakness

Principles for Operations

Normal:

- Use Runbooks for expected, repeatable tasks
- Prioritize tasks based on business impact
- Define escalation paths
- Enable push notifications
- Validate the achievement of outcomes and the effectiveness of KPIs and metrics
- Communicate status through dashboards
- For Workloads and Operations:
  - Define, collect, analyze, baseline metrics
  - Learn expected patterns of activity
  - Alert when outcomes are at risk
  - Alert when anomalies are detected

Cloud: These take on more significance in the Cloud

- Monitor Key Performance Indicators (KPI)
- Have a process per alert (Runbooks / Playbooks)
- Automate response to events

Operate - PROs

- Easy to build in infrastructure monitoring, alarms, and auto healing responses
- Remote access is built in for after-hours tasking
- No hardware repair downtime
- No hardware purchase lead time needed to grow your mission
- As personnel turnover, there is minimal training needed on infrastructure

Operate - CONs

- Daily/Weekly monitoring will be required
  - You will have to pay attention and analyze notices from the Cloud provider
- Loss of control of resource maintenance schedules
- Your applications must stay current – older hardware / software will be depreciated

Evolve Principles

Normal: Continuous improvement

- Capture changes to Runbooks as they get exercised
- Establish Playbooks as troubleshooting arises
- Establish a process for after-action or lessons-learned investigation and dissemination
- Analyze operating metrics and trends
- Periodically re-visit Key Performance Indicators (KPIs) for applicability and relevance
- Continuously train new personnel with GameDays and Runbook/Playbook training
- Make frequent, small, reversable changes

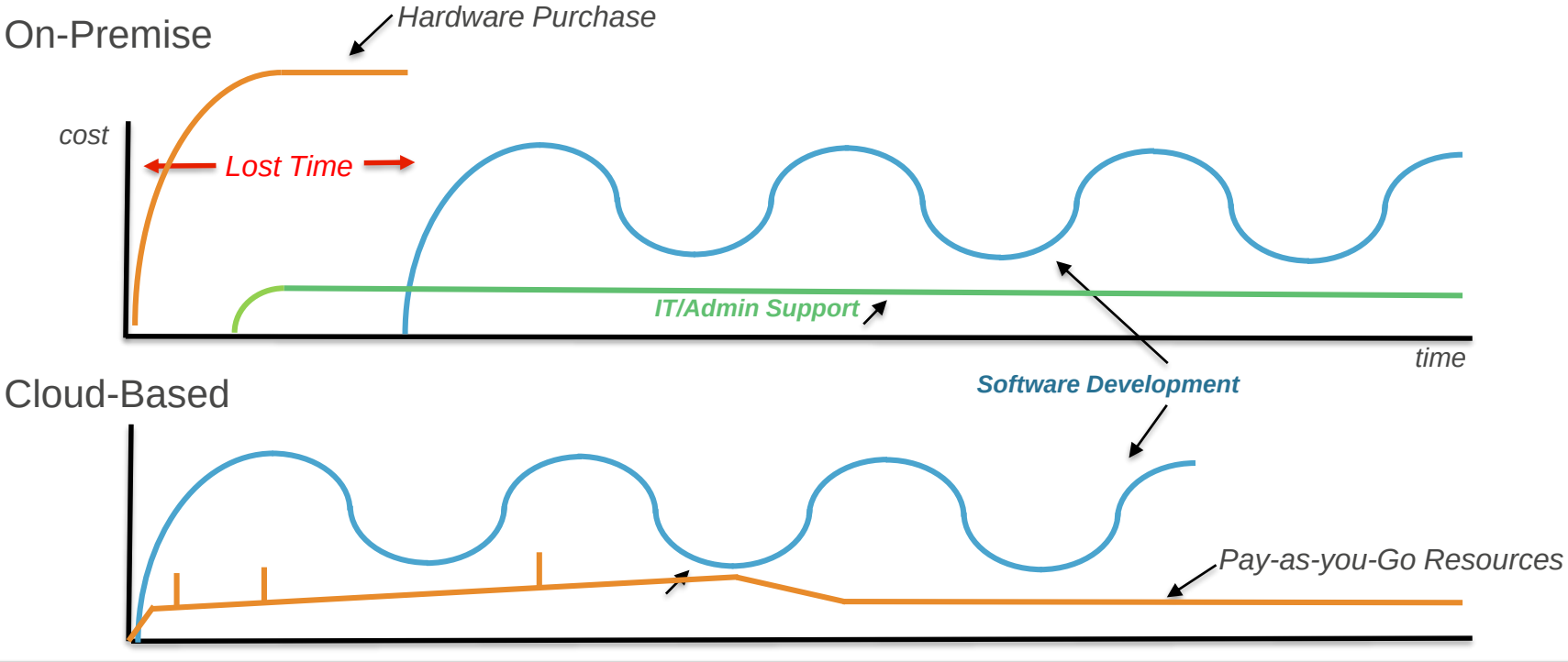
Cloud: Take advantage of low cost and two-way doors

- Experiment, Experiment, Experiment!
- Allocate human resources (allocate PEOPLE and labor HOURS) for evolving
- Implement / validate a potential solution rather than debate it
- Make sandboxes available to workloads for experimentation

Cost - PROs

- Pay as you Go: Spend, for what you use, is Worth it! Spend, for un-used, is Waste!
- Size (spend) for current resource need – auto scale as need increases
- Minimize up front purchases
- Lower cost allows for more experimenting and innovation
- Keeping costs down allows for you to put spend into critical (bottle neck) resources to improve your overall system
- Cost consciousness leads to adoption of many Cloud principles
- Consider Total Cost of Ownership (TCO) – space, HVAC, Electricity, guards, IT labor, re-capitalization, cost-of-money, etc. (next slide)

Cost During Development - TCO



Cost - CONs

- At Steady State, your ‘break even’ point for re-capitalization may be less than on-premise – depends on hours/month of resource usage
  - A case can be made to develop and operate in the cloud, until settled, then move back to on-premise (assumes no scaling needed!)
- Kids-in-a-Candy-Shop effect – it is WAY to easy to use more resources
  - Cost discipline will not happen on its own – management needs to be active
- Difficult to get to Apples-to-Apples comparison
  - Requires TCO which is often hidden
  - Attention will be drawn to the Cloud spend

Summary: Cloud vs. On-Premise

On-Premise	Cloud Approach
Research Up-Front	Continuous Discovery / Experimenting
Purchase All Possible Resources Up-Front	Pay-As-You-Go
Design for Max Usage	Design for Min Usage – Scale as Needed
Purchase Insurance Plan for Hardware	Design-in High Availability and Disaster Recovery – at almost no cost
Maintain a ‘Pet’ system	Build Cattle
Re-capitalize every 3+ years	Upgrade to Latest Services in Minutes/Hours – usually at a lower cost
Spend Time Patching / Upgrading	Focus on Mission with Managed Services

Now - what is the ‘Cloud’ To You?

Hopefully it is different ☺

Thank You