Compute as a Continuum
Continuum

• **What the app developer is responsible for**
  • **Before cloud (everything)**
    • Building or purchasing physical space
    • Purchasing and installing servers
    • Purchasing and configuring routers, switches, and network capacity
    • Purchasing and installing operating system software, middleware, databases
    • Manually scaling all of the above based on load
    • Managing all data backups
    • Implementing all authentication and data analytics functionality
  • **Now**
    • Everything (pretend the cloud doesn't exist)
    • Some (DevOps, Hybrid Cloud)
    • None (NoOps)
    • Hint: prefer None 😊
Infrastructure-as-a-Service

- IaaS
  - Hardware as a service
    - On-demand processing power and storage
  - Server functions and software managed manually by the app developer
    - Logging, operating system configuration, library installation, patching, security, analytics, load balancing, replication, backups, etc.
  - How to manage huge IaaS deployments?
    - Chef, Puppet, Ansible, Cloud Deployment Manager
    - Deploy collections of VMs that implement application as a service
Containers-as-a-Service

- OS images as a service
- Manage collections of containers that implement application as a service
- Often mapped onto VMs in IaaS
Platform-as-a-Service

- **PaaS**
  - Cloud provides development environment to application
  - Environment automatically scales infrastructure based on application usage
- **NoOps**
  - Just supply the app
- **Issues**
  - Vendor lock-in
  - Level of control over scaling
  - Fixed platform environment that app must adapt to
- **Analogy**
  - IaaS like a stick-shift
  - PaaS like an automatic

(AWS)
Functions-as-a-Service

- Serverless computing
  - Special case of PaaS where infrastructure brought up for execution of a single function
  - Ephemeral servers (5 minute max on AWS Lambda)
Software-as-a-Service

• SaaS
  • App written and hosted completely by provider
  • Software rented on a subscription basis (or for free)
    • Includes software, hardware and support
  • Outsources hosting of apps
  • Ranges from simple (storage) to complex (productivity apps, data analytics)
How to choose

- How much control do you need?
  - PaaS
    - Not flexible
    - NoOps
  - IaaS
    - Flexible but with Ops overhead
    - Requires manual management
How to choose

- Note use of containers throughout so you can migrate down easily

Abstraction - what do you want to think about?

- Cloud Functions
  - Events
  - Function definitions

- App Engine
  - Code
  - HTTP Requests

- Container Engine
  - Applications, not computers or containers
  - What programs? How are they connected? State?

- Compute Engine
  - Your software, Operating system / disk images
  - CPU, RAM, Disk
  - Networking: Firewall rules, Load balancers, VPNs
Technical requirements pull you down the stack

Cloud Functions

App Engine

Container Engine

Support any programming language, run in containers

Hybrid, need specific OS, network protocols beyond HTTP/S

GPUs, need specific kernel, Windows, software licensing requirements, migrating most existing systems.
Team and organization

- Cloud Functions
  - Team is mostly dev focused

- App Engine
  - Team is mostly dev focused

- Container Engine
  - Team integration: Dev, Ops, Security work together, Org is open to app architecture updates

- Compute Engine
  - Adaptable to various team structures and tool preferences.