Cloud service models

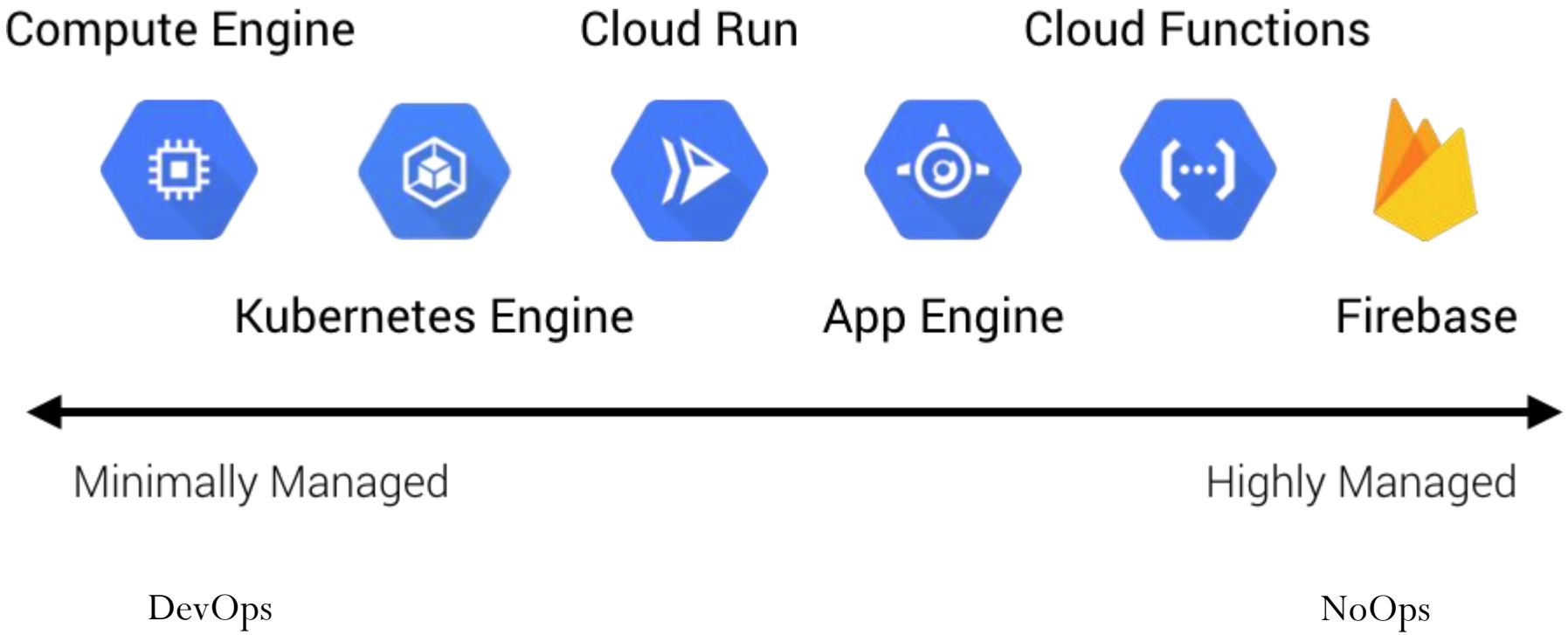
Portland State
Computer Science

Flexible service and management models

- Cloud providers support management style you prefer
- Before cloud, one model
  - Purchase building, hardware, routers, switches, network, operating system software, database software, middleware, etc.
  - Manually scale based on load
  - Manually perform data backups
  - Implement all authentication and data analytics functionality
- With cloud, a choice!
  - Client does everything (pretend the cloud provides nothing more than a virtual data center)
  - Client does some (DevOps, Hybrid Cloud)
  - Client does none (NoOps, fully managed)
  - Hint: prefer None ☺

Mirrors the offerings

- Levels of abstraction and control

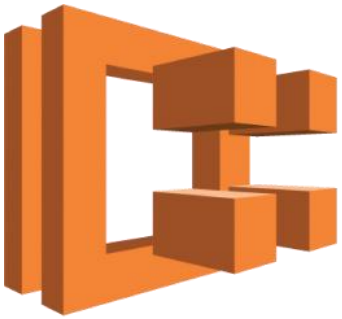


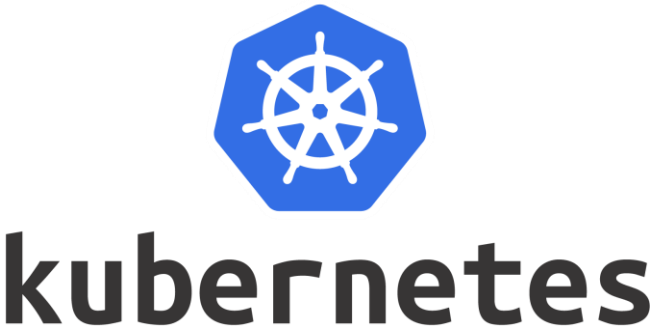
Infrastructure-as-a-Service

- 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, replication, backups, etc.
    - How to manage huge IaaS deployments?
      - Chef, Puppet, Ansible, Terraform, Cloud Deployment Manager
      - Deploy collections of VMs that implement application

Container orchestration services

- Manage collections of containers that implement application as a service
  - Containers often mapped onto VMs in IaaS
  - Orchestration run-time keeps infrastructure running properly
- Typically requires at least one server to be running
  - But, want something that costs $0 if not used!



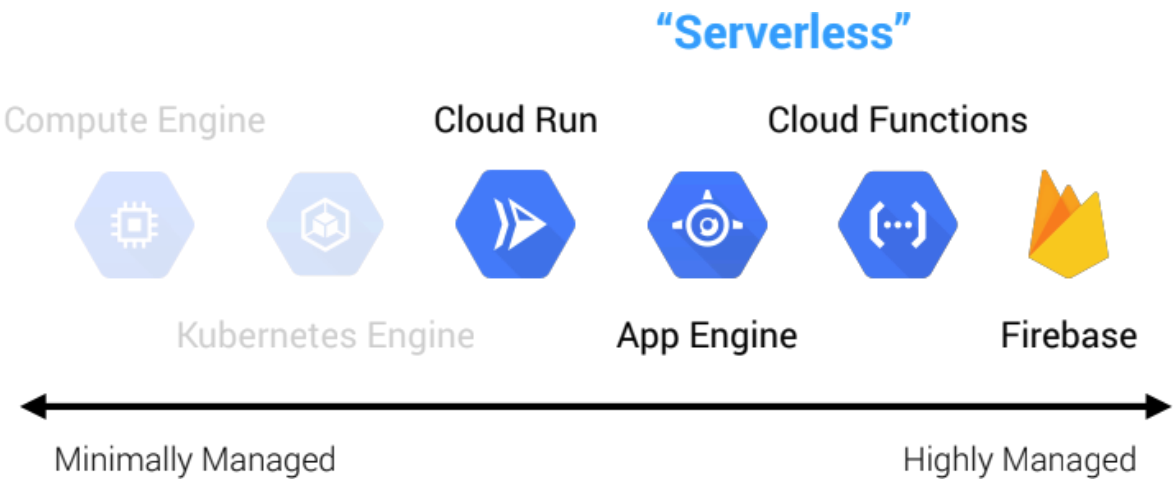


Serverless computing

- A solution that costs nothing if nobody is using it
  - No up front provisioning
  - No management of servers
  - Pay for what you use
  - Can go down to 0 servers and "wake-up" when needed
- Model
  - Developer produces code for application
  - Infrastructure provided automatically to run it
  - No machines or networks exposed to the developer
  - Analogy
    - IaaS like a stick-shift
    - Serverless like an automatic

Serverless computing

- Many incarnations
  - Serverless containers (CloudRun)
  - Serverless platforms (AppEngine)
  - Serverless functions (Cloud Functions)
  - Serverless backends and applications (Firebase)
  - Serverless data storage (Google Cloud Storage)
  - Serverless databases and data warehouses (Cloud Datastore, BigQuery)



Serverless containers

- User supplies container image
  - Cloud dynamically instantiates container on a server on-demand
  - Scales copies of containers up when demand is high
  - Removes all running containers if no demand
  - Just supply the container (abstract out machines)
  - No vendor lock-in
  - Issues
    - Level of control over scaling

Serverless platforms

- aka Platform-as-a-Service
  - Cloud provides development environment (pre-defined containers) to application
  - Automatically scales containers up and down based on application usage
  - Just supply the app (abstract out machines and container)
    - No need to learn Docker! ☺
  - Issues
    - Level of control over scaling
    - Vendor lock-in
    - Fixed platform environment that app must adapt to

CLOUDFOUNDRY

Google app engine

heroku

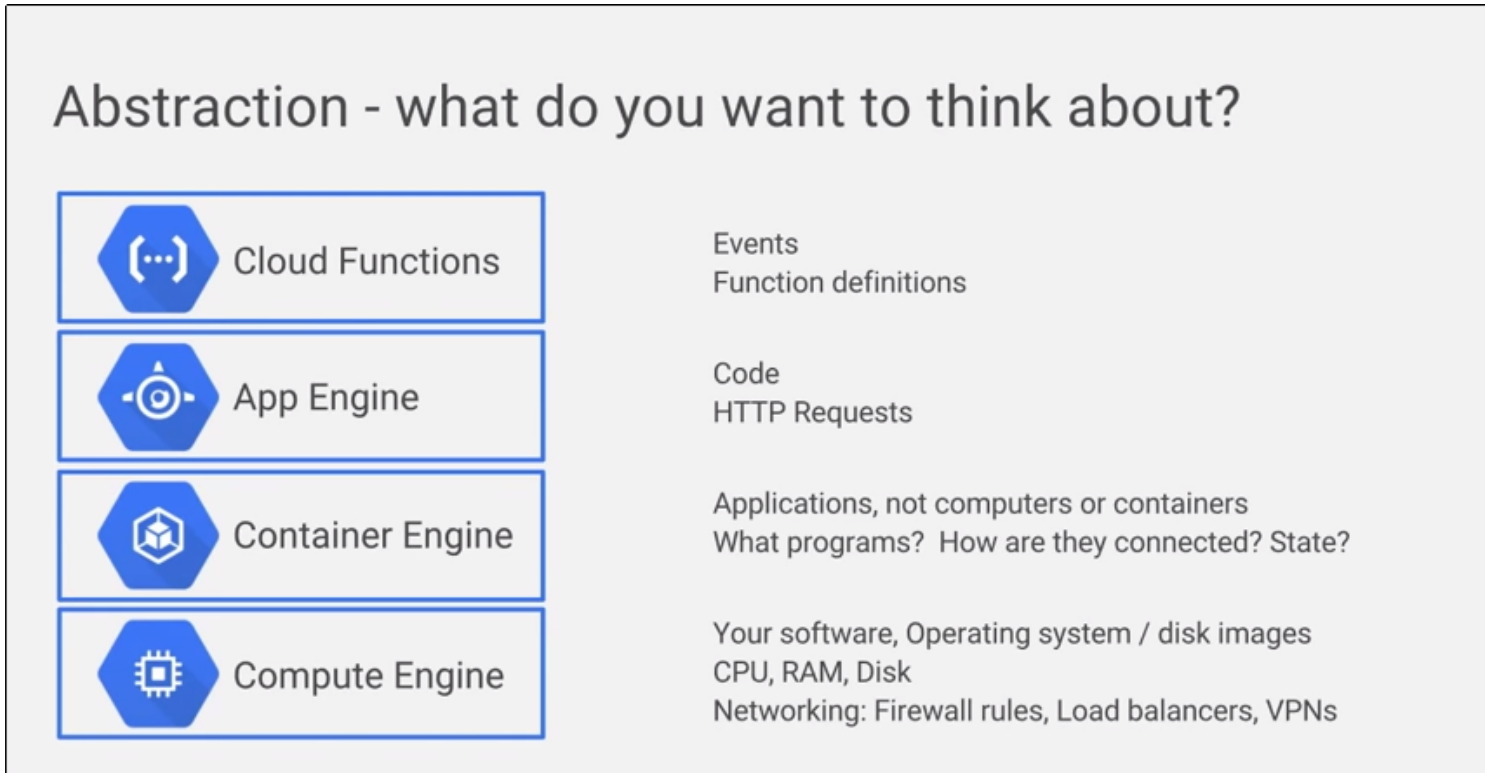
(AWS)

Serverless functions

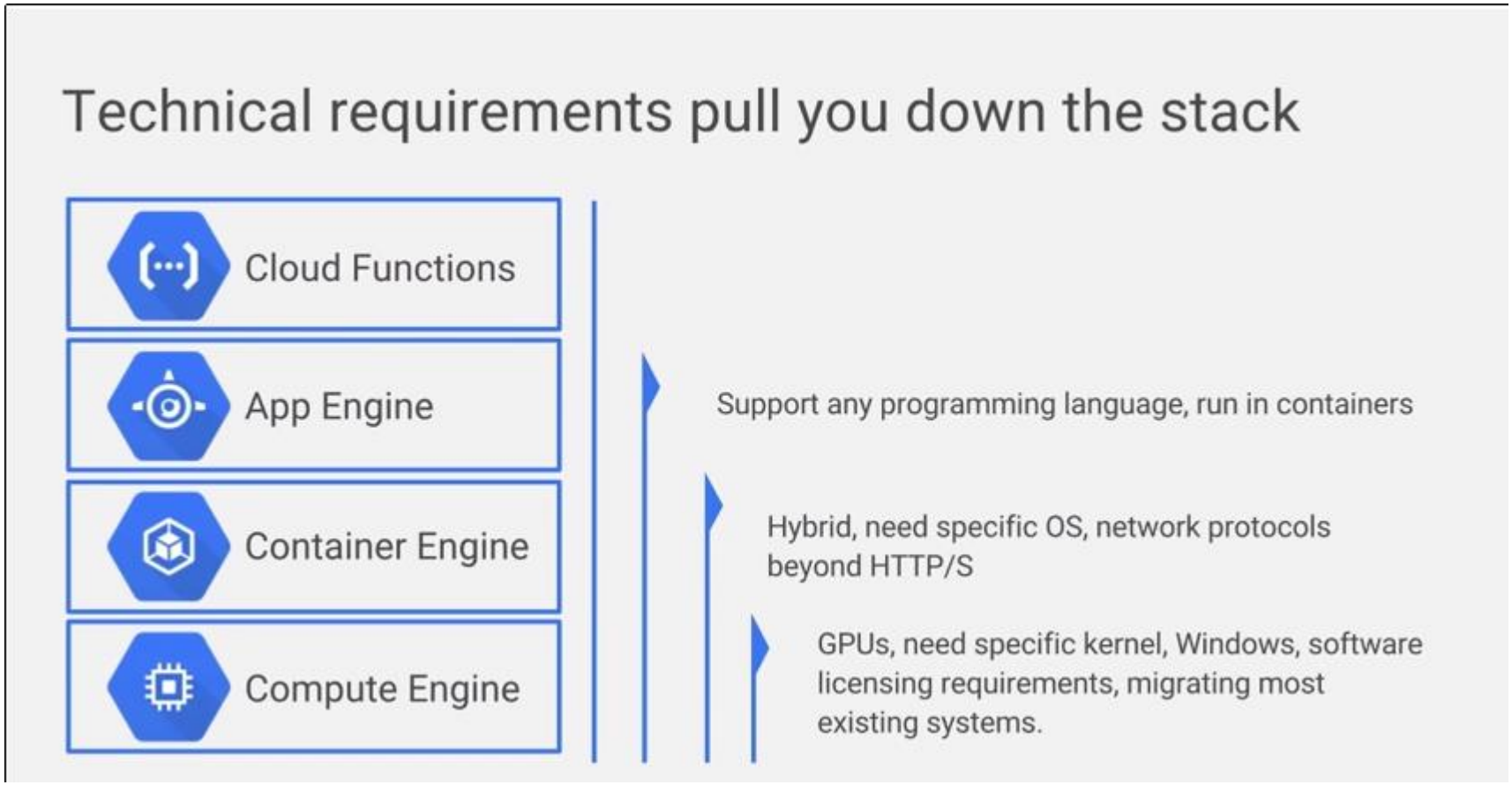
- aka Functions as a service
  - Special case of PaaS where infrastructure brought up for execution of a single function
  - Enables "event-driven" computing
    - Single-purpose function executed in response to some asynchronous event
    - Run on ephemeral, pre-defined, run-time systems (e.g. containers)
    - Typically stateless
      - Any state that needs to be stored written out to managed data stores
  - Ephemeral servers (5 minute max on AWS Lambda)
    - AWS Lambda, Cloud Functions, Azure Functions

How to choose

- How to choose?
  - https://dzone.com/articles/serverless-on-gcp
  - https://medium.com/google-cloud/cloud-run-vs-cloud-functions-whats-the-lowest-cost-728d59345a2e
- Application's abstraction level



- Application's run-time environment requirements



- Employee expertise