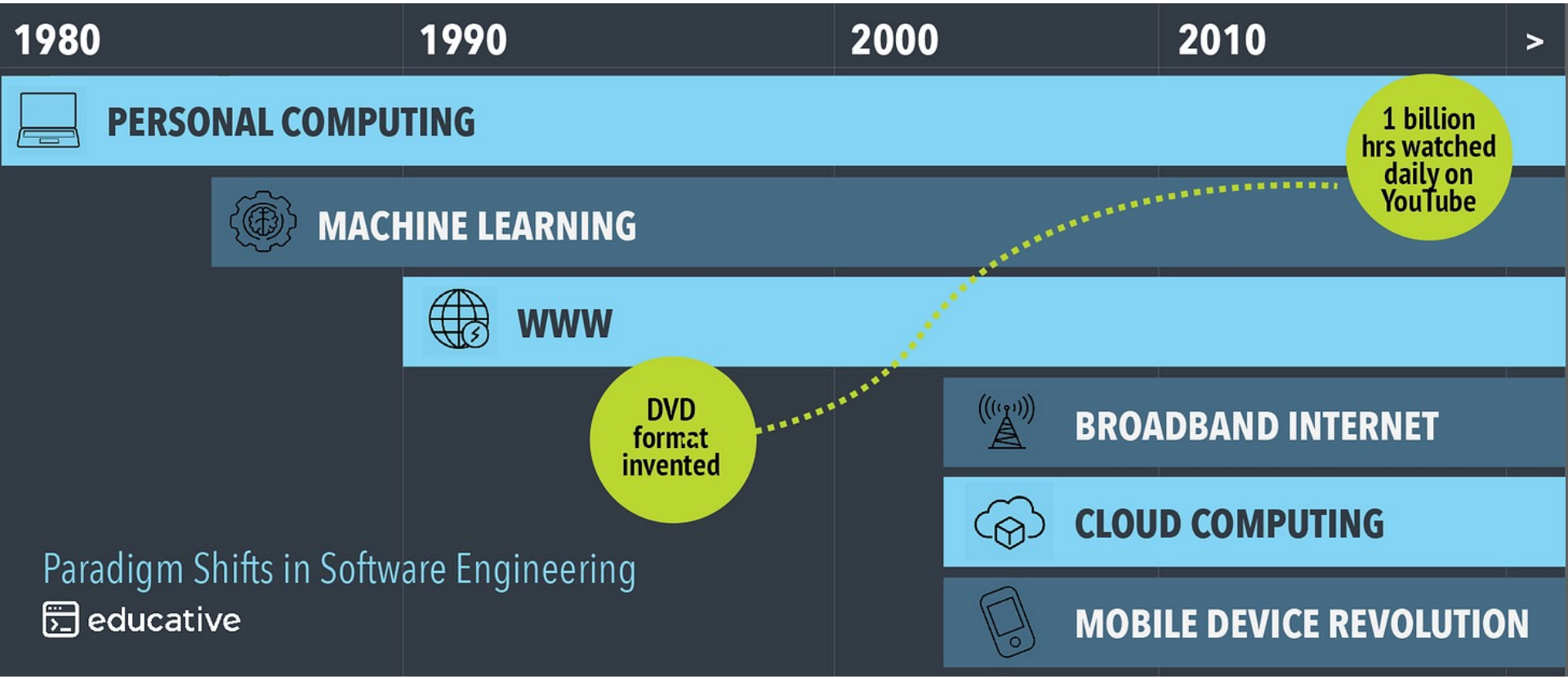
UC San Diego

DSC 102
Systems for Scalable Analytics

Haojian Jin

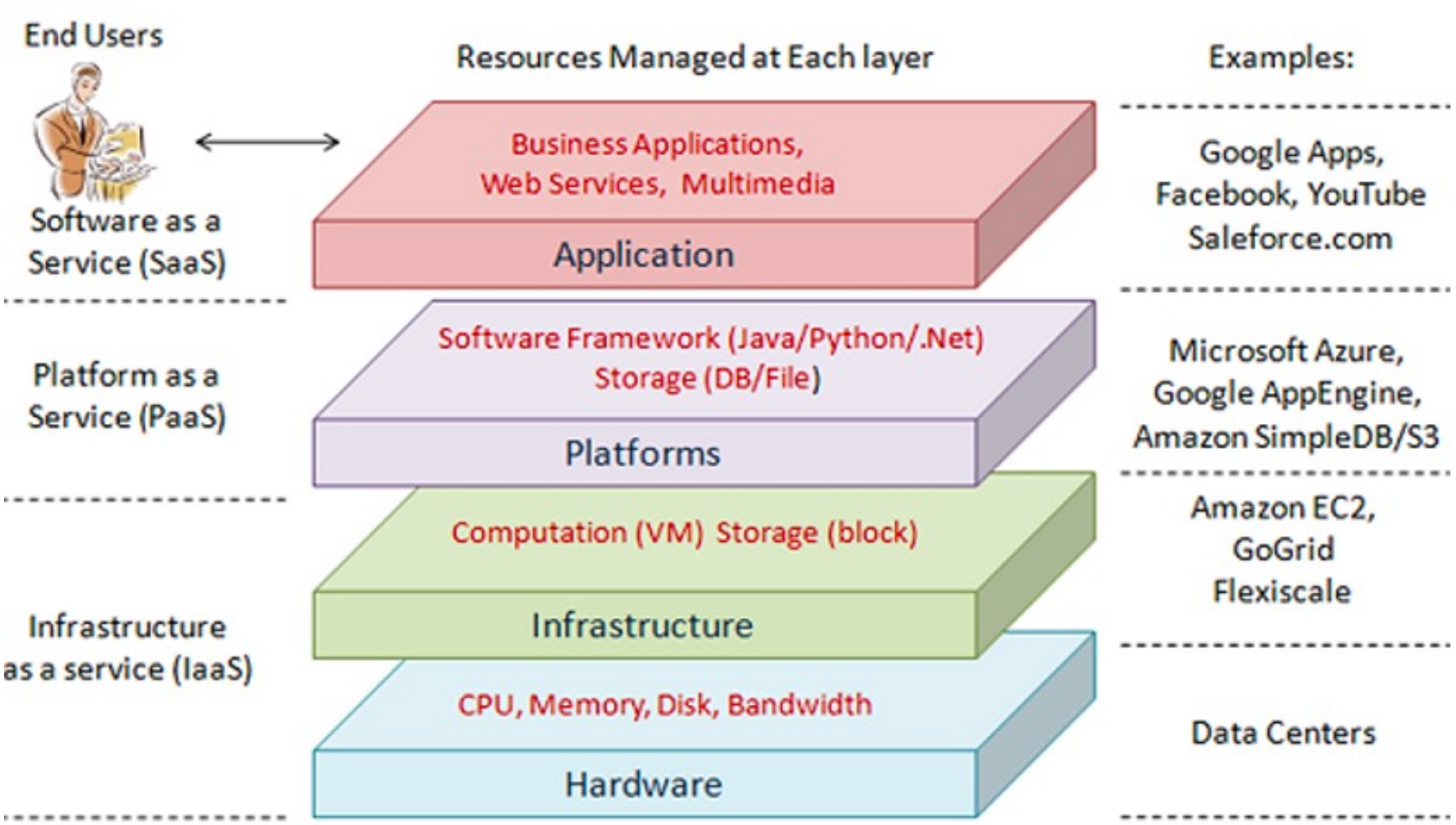
Topic 2: Basics of Cloud Computing


1980
1990
2000
2010
>
PERSONAL COMPUTING
MACHINE LEARNING
WWW
BROADBAND INTERNET
CLOUD COMPUTING
MOBILE DEVICE REVOLUTION
1 billion hrs watched daily on YouTube
DVD format invented
Paradigm Shifts in Software Engineering
educative

Cloud Computing

- Compute, storage, memory, networking, etc. are virtualized and exist on *remote servers*; *rented* by application users
- Main pros of cloud vs on-premise clusters:
  - **Manageability**: Managing hardware is not user's problem
  - **Pay-as-you-go**: Fine-grained pricing economics based on actual usage (granularity: seconds to years!)
  - **Elasticity**: Can dynamically add or reduce capacity based on actual workload's demand
- Infrastructure-as-a-Service (IaaS); Platform-as-a-Service (PaaS); Software-as-a-Service (SaaS)

Cloud Computing



https://benchpartner.com/layered-model-of-cloud-computing

Examples of AWS Cloud Services

- **IaaS**:
  - **Compute**: EC2, ECS, Fargate, Lambda
  - **Storage**: S3, EBS, EFS, Glacier
  - **Networking**: CloudFront, VPC
- **PaaS**:
  - **Database/Analytics Systems**: Aurora, Redshift, Neptune, ElastiCache, DynamoDB, Timestream, EMR, Athena
  - **Blockchain**: QLDB; **IoT**: Greengrass
- **SaaS**:
  - **ML/AI**: SageMaker, Elastic Inference, Lex, Polly, Translate, Transcribe, Textract, Rekognition, Ground Truth
  - **Business Apps**: Chime, WorkDocs, WorkMail

Evolution of Cloud Infrastructure

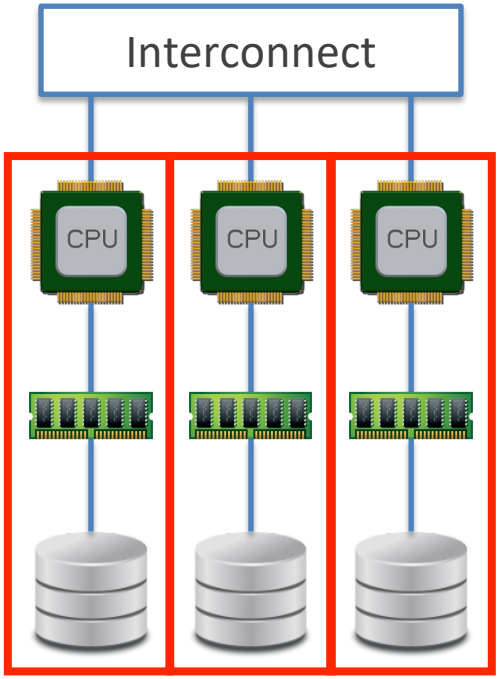
- **Data Center**: Physical space from which a cloud is operated
- **3 generations of data centers/clouds:**
  - **Cloud 1.0 (Past)**: Networked servers; user rents servers (time-sliced access) needed for data/software
  - **Cloud 2.0 (Current)**: "Virtualization" of networked servers; user rents amount of resource capacity; cloud provider has a lot more flexibility on provisioning (multi-tenancy, load balancing, more elasticity, etc.)
  - **Cloud 3.0 (Ongoing Research)**: "Serverless" and disaggregated resources all connected to fast networks

A brief history

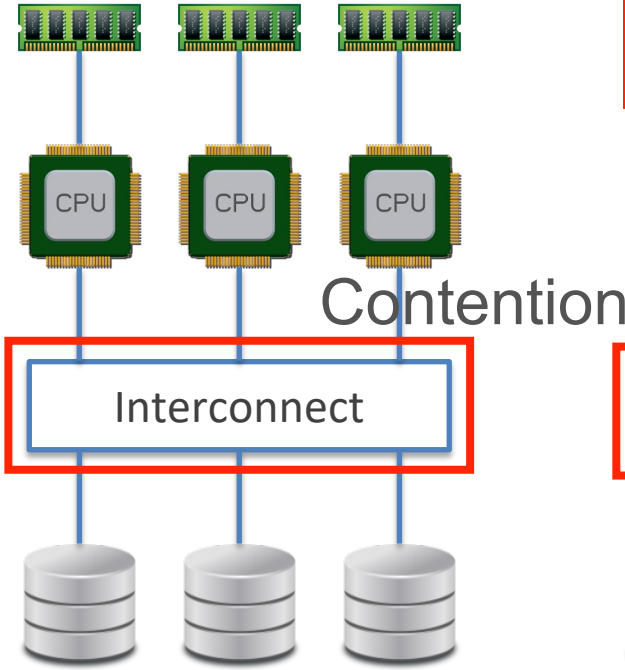
- **Data Center**: Physical space from which a cloud is operated
- **3 generations of data centers/clouds:**
  - **Cloud 1.0 (Past)**: Networked servers; user rents servers (time-sliced access) needed for data/software
  - **Cloud 2.0 (Current)**: “Virtualization” of networked servers; user rents amount of resource capacity; cloud provider has a lot more flexibility on provisioning (multi-tenancy, load balancing, more elasticity, etc.)
  - **Cloud 3.0 (Ongoing Research)**: “Serverless” and disaggregated resources all connected to fast networks

3 Paradigms of Multi-Node Parallelism

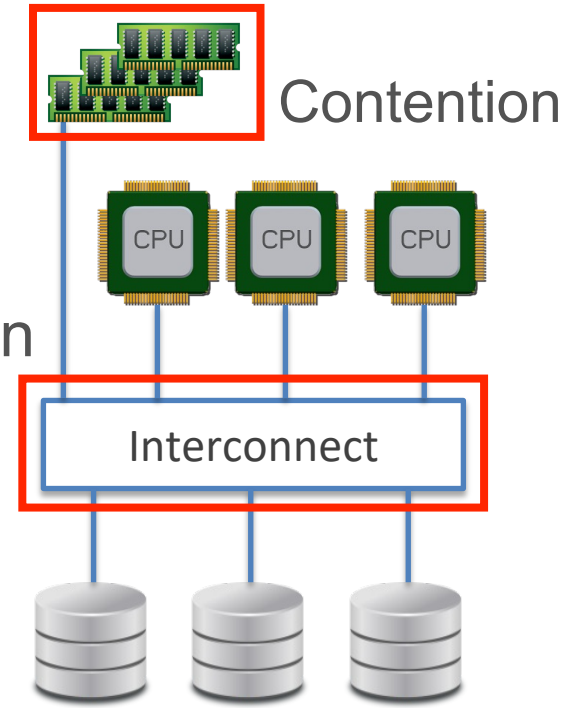
Independent Workers



Shared-Nothing Parallelism



Shared-Disk Parallelism



Shared-Memory Parallelism

Most parallel RDBMSs (Teradata, Greenplum, Redshift), Hadoop, and Spark use shared-nothing parallelism

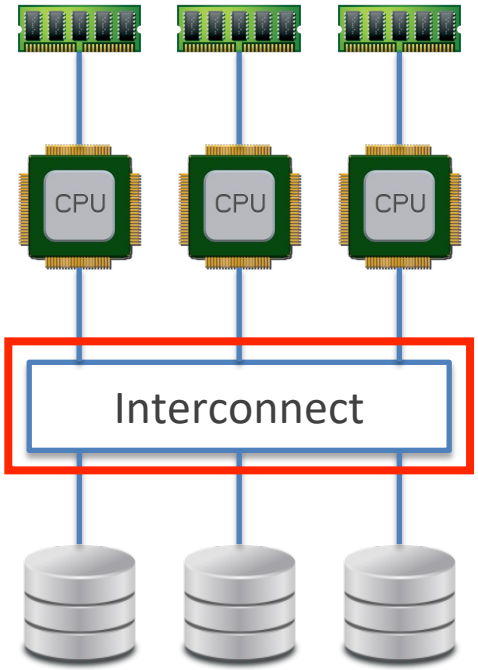
What Makes Snowflake So Powerful — It's the Hybrid of Shared Disk and Shared Nothing Architecture

A Arun Kaushik · Follow
6 min read · Feb 14, 2020

The IT industry has been undergoing a massive transformation for last few years. Companies have been heavily investing in upgrading their technology stack to reduce operational costs and offer a better service to customers. Digital, Cloud, As-a-Service etc are some of the buzz words which have been driving transformation projects across the industry.

https://medium.com/@a.kaushik5587/what-makes-snowflake-so-powerful-its-the-hybrid-of-shared-disk-and-shared-nothing-architecture-5b4fa8f039fa

Revisiting Parallelism in the Cloud

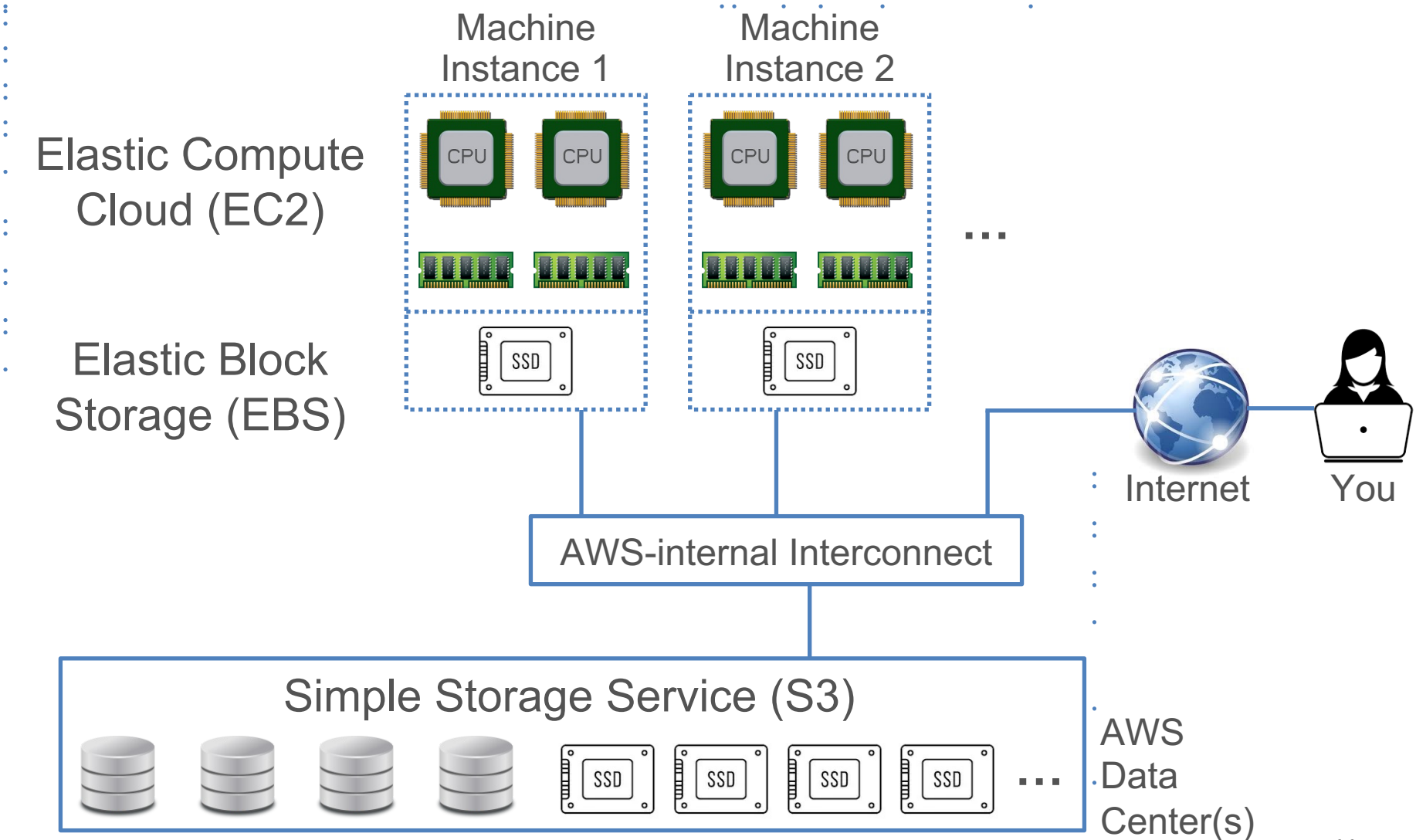


Shared-Disk Parallelism

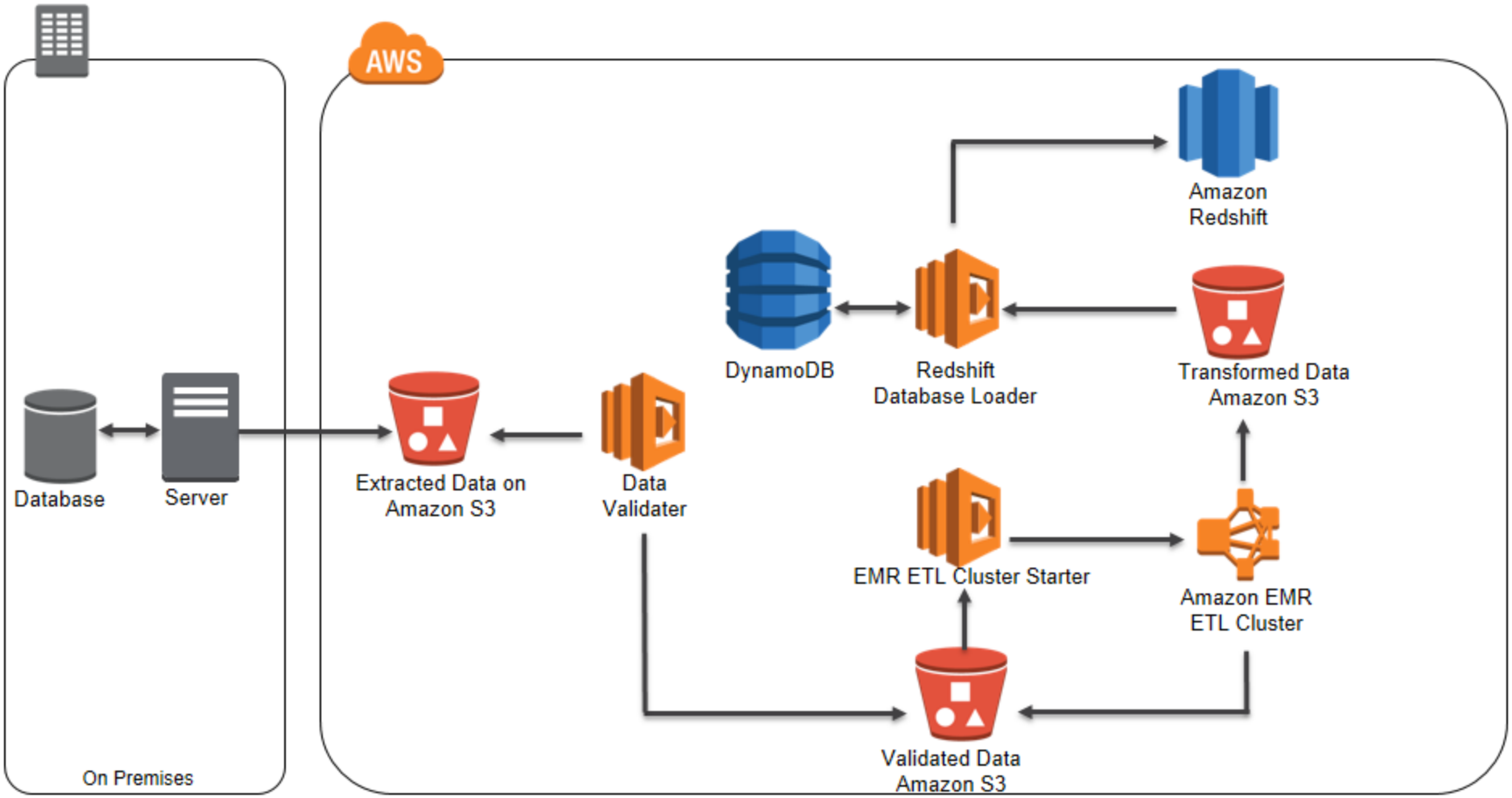
Modern networks in data centers have become much faster: 100GbE to even TbE!

- **Decoupling** of compute+memory from storage is common in cloud
  - *Hybrids* of shared-disk parallelism + shared-nothing parallelism
  - E.g, store datasets on S3 and read as needed to local EBS

Example: AWS Services for PA1

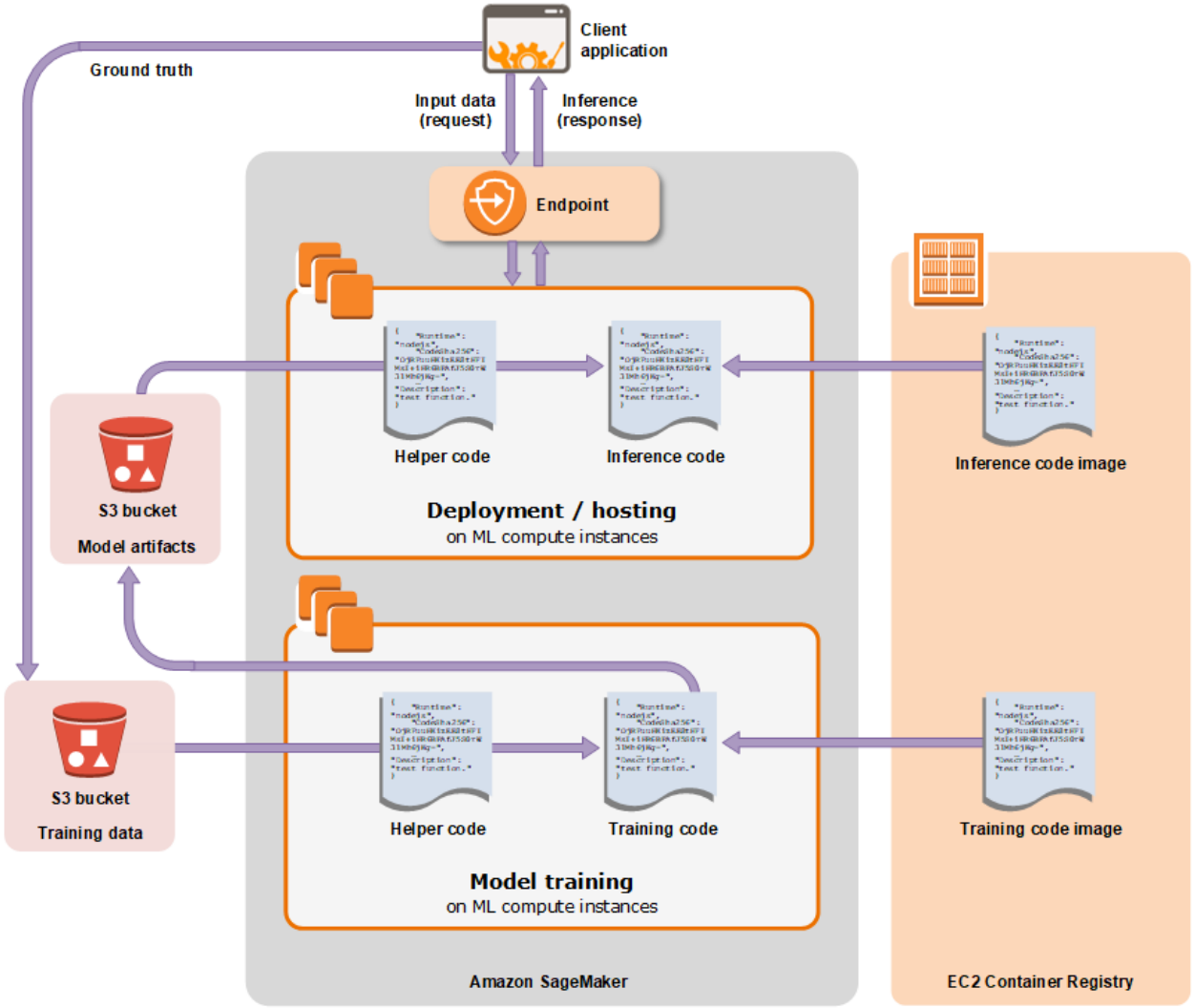


Example: AWS DB/Analytics Services



https://aws.amazon.com/blogs/big-data/automating-analytic-workflows-on-aws/

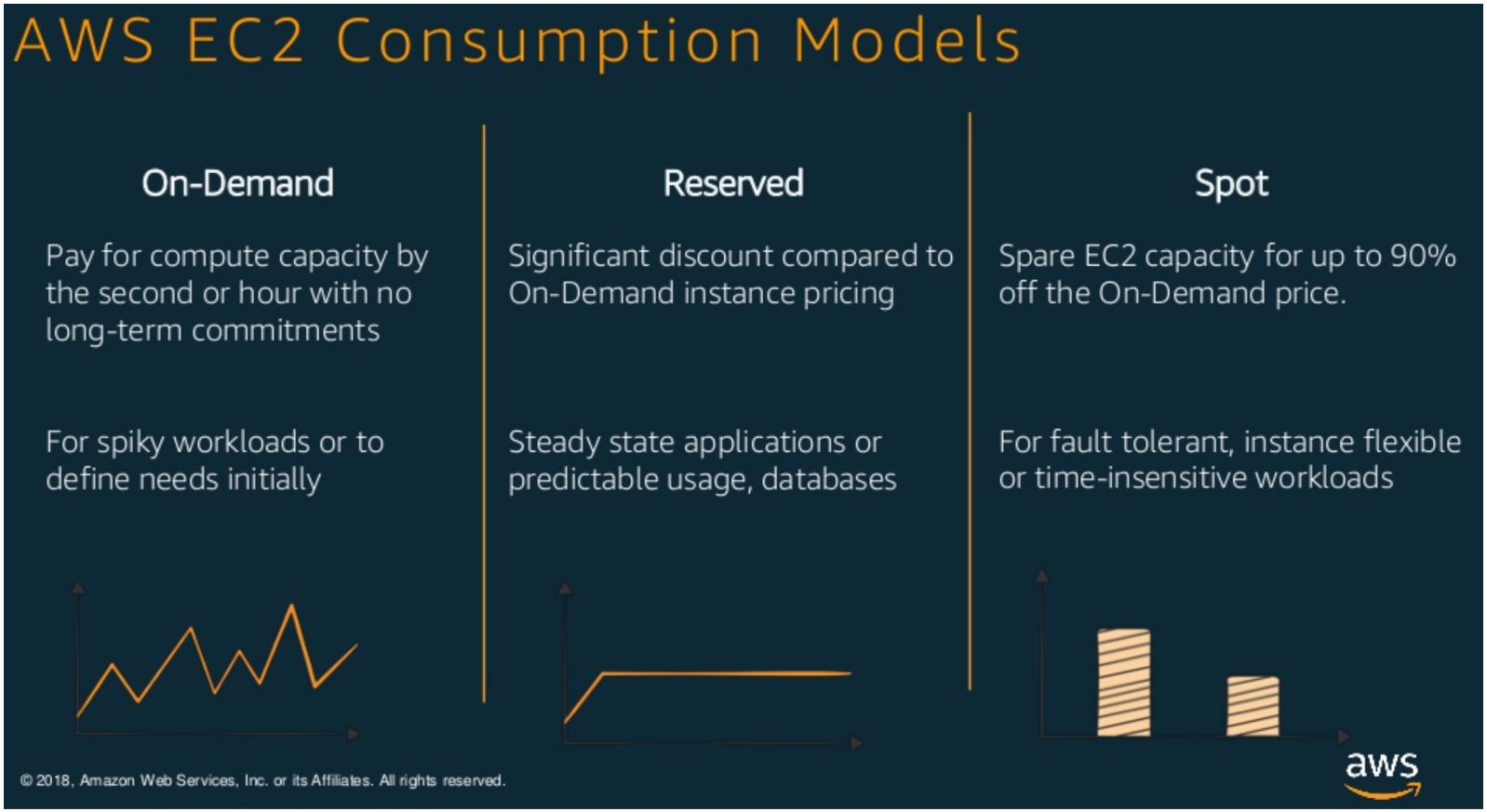
Example: AWS ML Services



https://docs.aws.amazon.com/sagemaker/latest/dg/how-it-works-training.html

New Cloud Renting Paradigms

- Cloud 2.0's flexibility enables radically different paradigms
- AWS example below; Azure and GCP have similar gradations



https://www.slideshare.net/AWSUsersGroupBengalu/amazon-ec2-spot-instances

More on Spot vs On-Demand

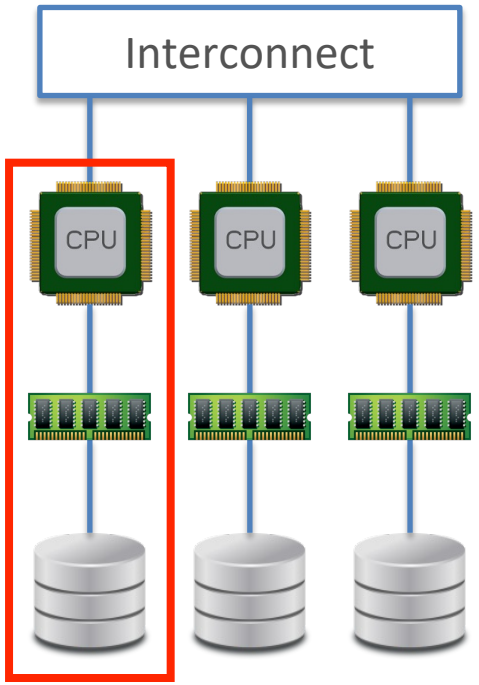
	Spot Instances	On-Demand Instances
Launch time	Can only be launched immediately if the Spot Request is active and capacity is available.	Can only be launched immediately if you make a manual launch request and capacity is available.
Available capacity	If capacity is not available, the Spot Request continues to automatically make the launch request until capacity becomes available.	If capacity is not available when you make a launch request, you get an insufficient capacity error (ICE).
Hourly price	The hourly price for Spot Instances varies based on demand.	The hourly price for On-Demand Instances is static.
Rebalance recommendation	The signal that Amazon EC2 emits for a running Spot Instance when the instance is at an elevated risk of interruption.	You determine when an On-Demand Instance is interrupted (stopped, hibernated, or terminated).
Instance interruption	You can stop and start an Amazon EBS-backed Spot Instance. In addition, the Amazon EC2 Spot service can interrupt an individual Spot Instance if capacity is no longer available, the Spot price exceeds your maximum price, or demand for Spot Instances increases.	You determine when an On-Demand Instance is interrupted (stopped, hibernated, or terminated).

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html

New Cloud Renting Paradigms

Such bundling means some applications might under-utilize some resources!

- **Serverless** paradigm gaining traction for some applications, e.g., online ML prediction serving on websites
- User gives a program (function) to run and specifies CPU and DRAM needed
- Cloud provider abstracts away all resource provisioning entirely
- Higher resource efficiency; much cheaper, often by 10x vs Spot instances
- Aka *Function-as-a-Service* (FaaS)



Shared-Nothing Parallelism

Car Analogy for Serverless Cloud

Own a car
(Bare metal servers)

Rent a car
(VPS)

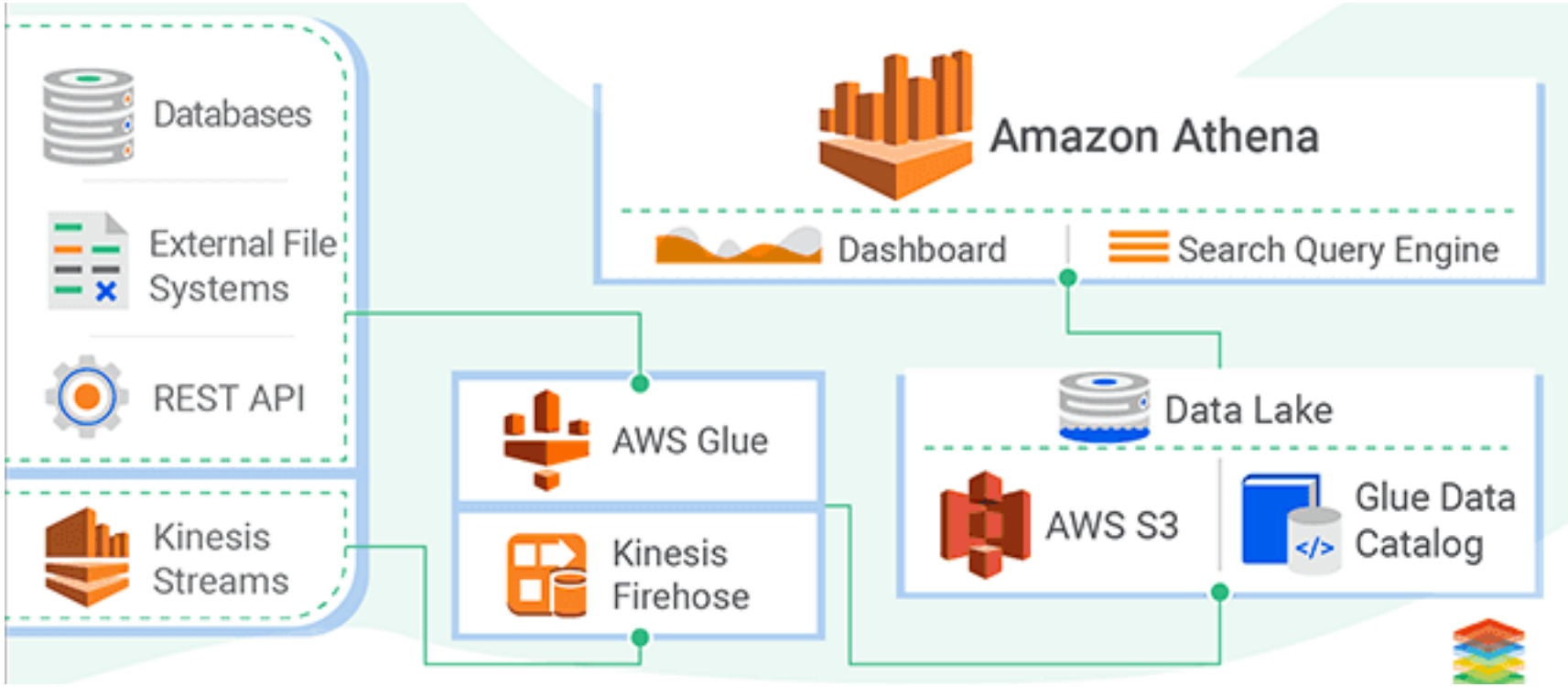
City car-sharing
(Serverless)

Cars are parked **95%** of the time (loige.link/car-parked-95)

How much do you use the car?

https://www.slideshare.net/loige/building-a-serverless-company-with-nodejs-react-and-the-serverless-framework-jsday-2017-verona

Example: Serverless RDBMS on AWS



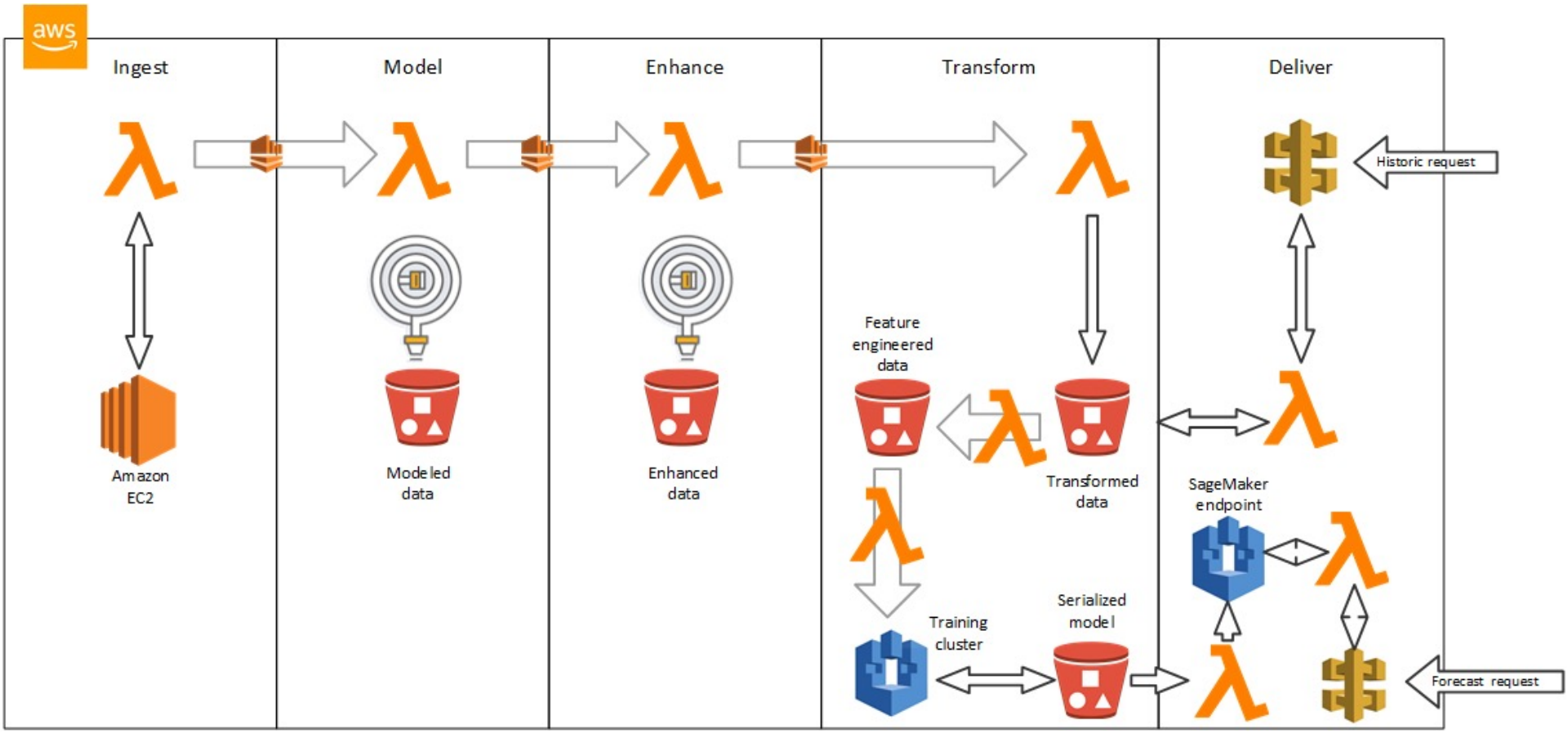
Remote read of data from S3

Schema-on-read
Many data formats

Simple interactive queries

https://www.xenonstack.com/blog/amazon-athena-quicksight/

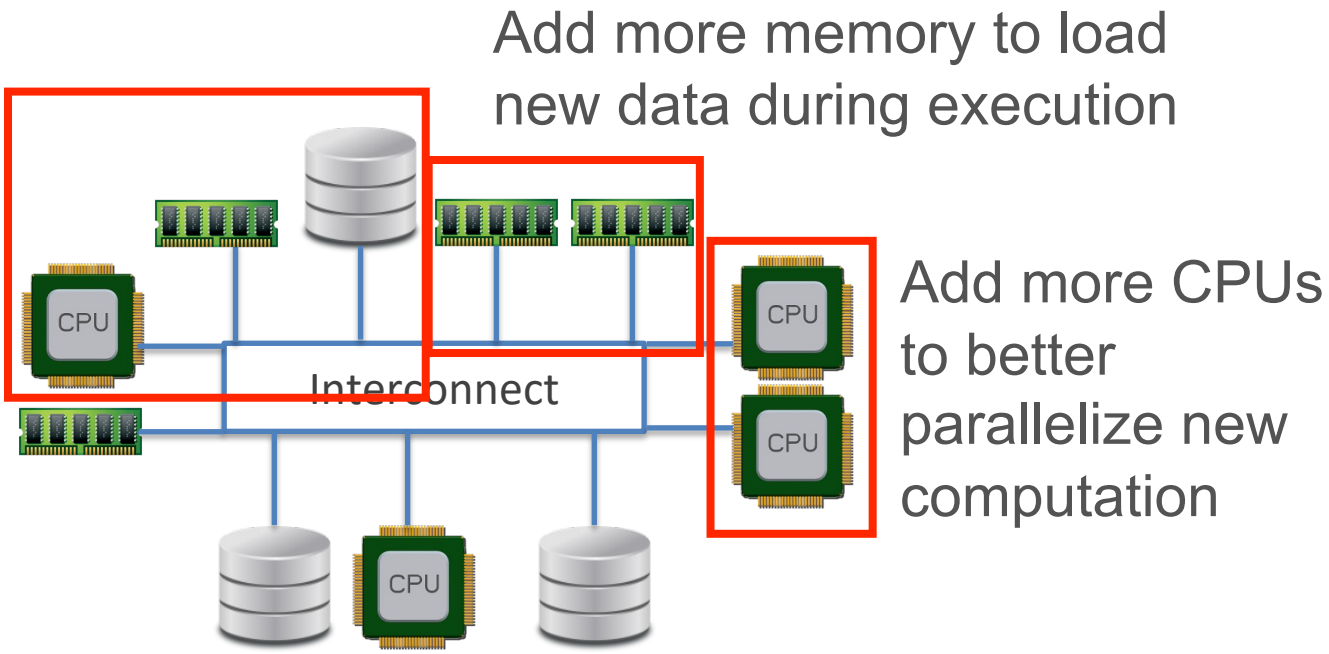
Example: Serverless ML app. on AWS



https://aws.amazon.com/quickstart/architecture/predictive-data-science-sagemaker-and-data-lake/

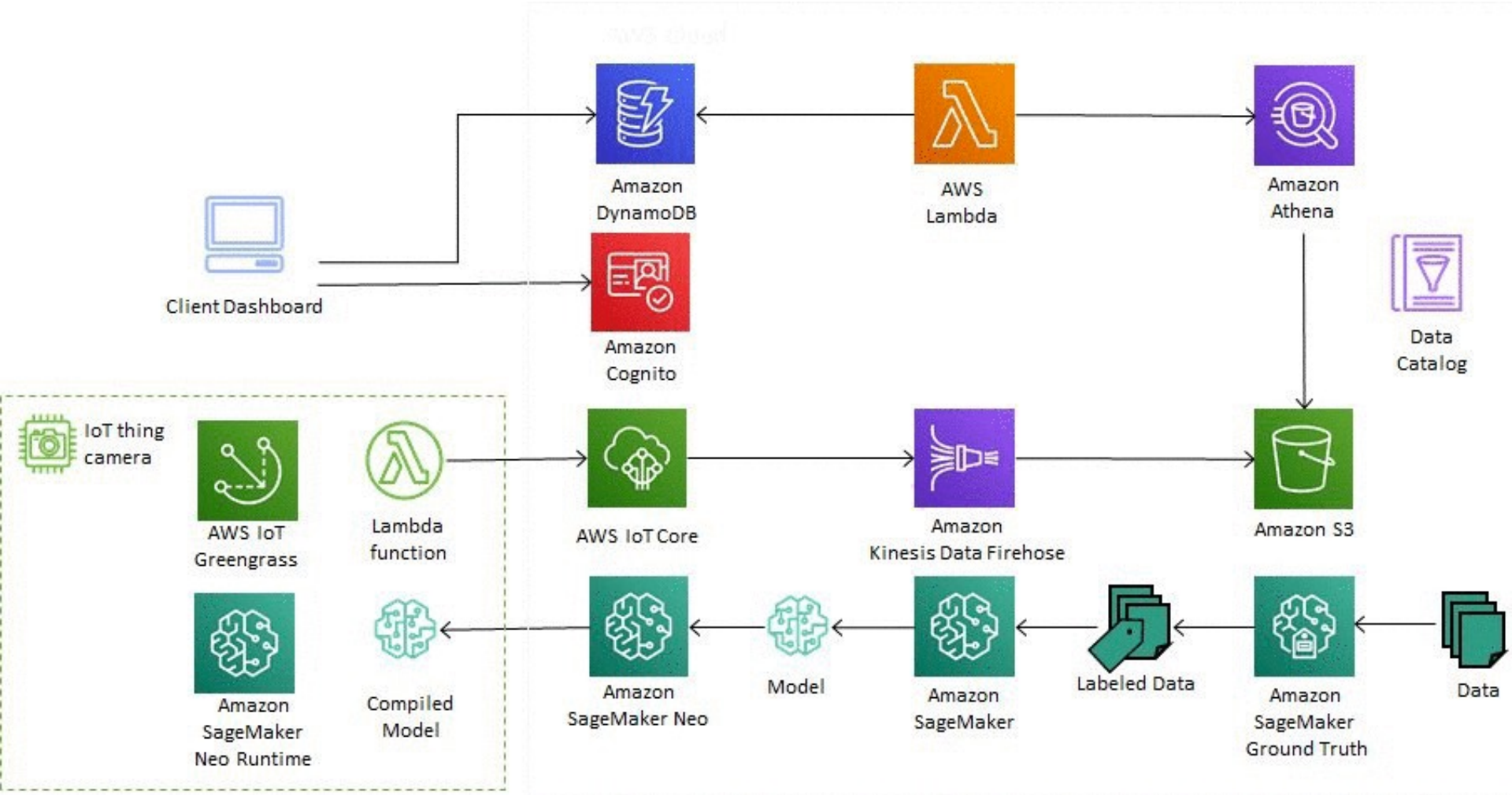
Disaggregation: Glimpse into the Future?

- Logical next step in serverless direction: full **resource disaggregation**! That is, compute, memory, storage, etc. are all network-attached and elastically added/removed



Ongoing Research: Fulfill this promise with low latency!

Example: AWS services for IoT app.



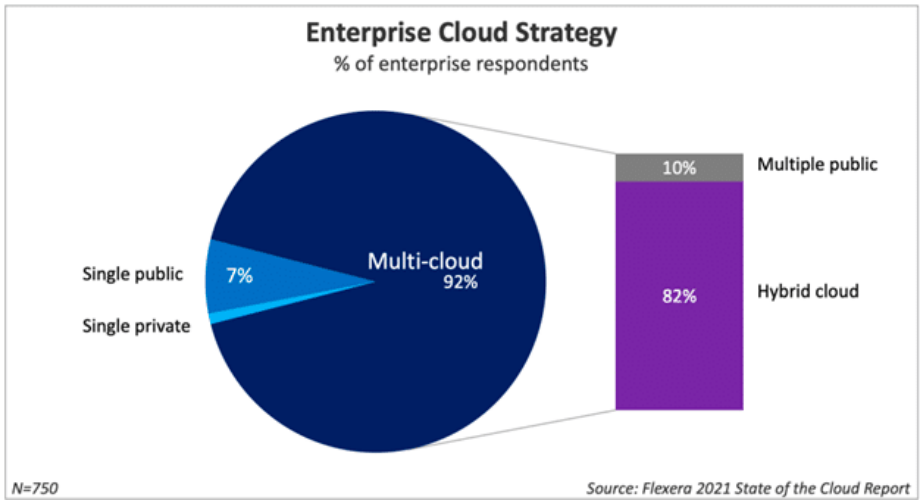
https://aws.amazon.com/blogs/machine-learning/amazon-sagemaker-neo-helps-detect-objects-and-classify-images-on-edge-devices/

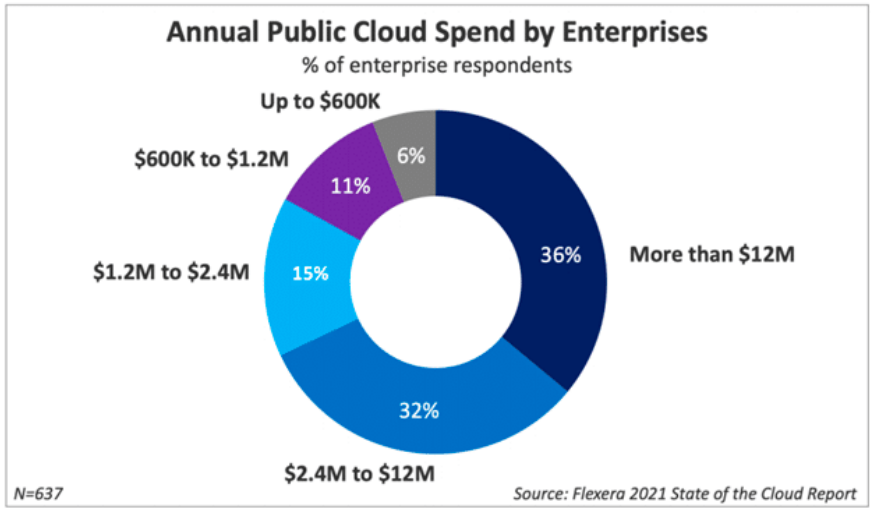
OMG, is all this complexity worth it?!

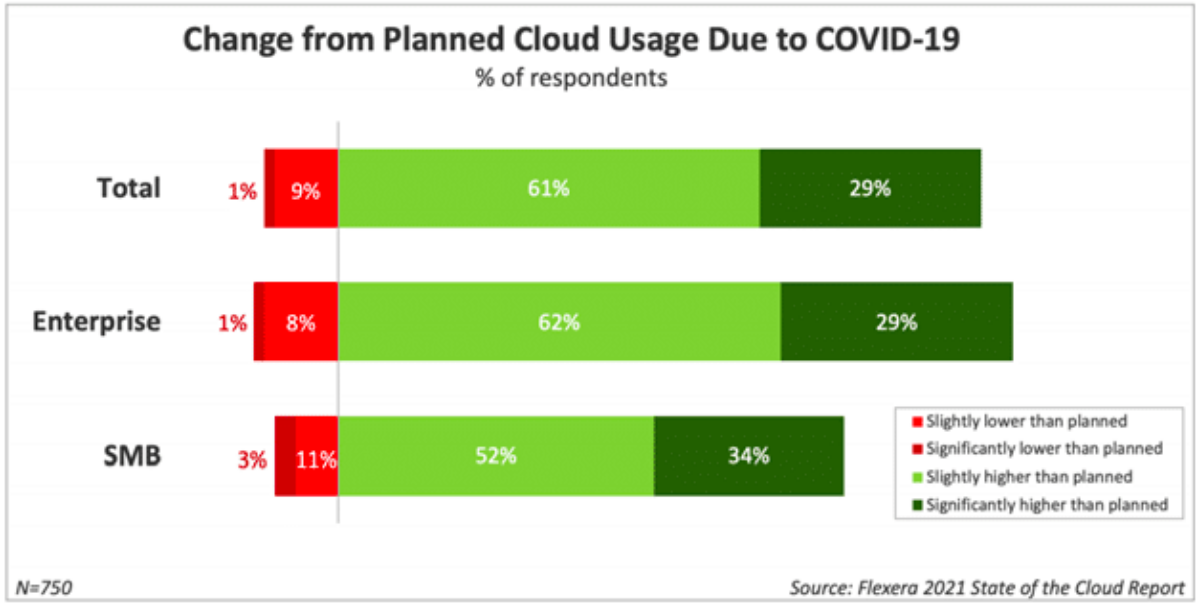
- Depends on user's/application's Pareto tradeoffs! :)
- **On-premise** cluster are still common in large enterprises, healthcare, and academia; "hybrid clouds" too
- Recall main pros of cloud: manageability, cost, and elasticity
- Some main cons of cloud (vs on-premise):
  - **Complexity** of composing cloud APIs and licenses; data scientists must keep relearning; "CloudOps" teams
  - **Cost** over time can *crossover* and make it costlier!
  - Easier to **waste money** accidentally on the fly
  - **"Lock-in"** by cloud vendor
  - **Privacy**, **security**, and **governance** concerns
  - **Internet disruption** or **unplanned downtime**, e.g., AWS outage in 2015 made Netflix, Tinder, etc. unavailable! :)

OMG, is all this complexity worth it?!

The State of the Cloud Survey

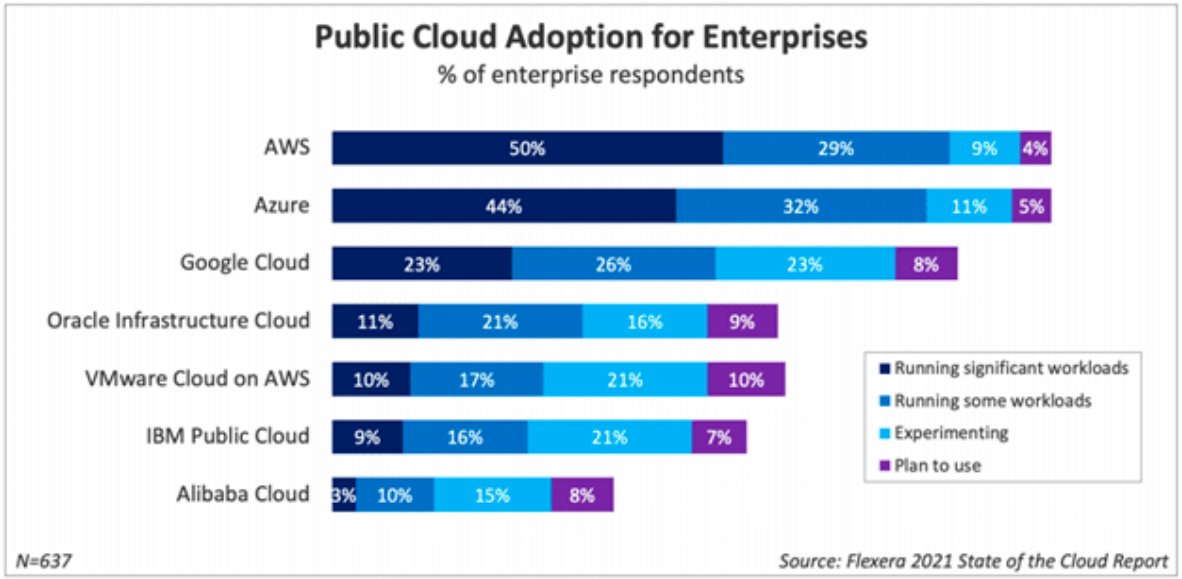


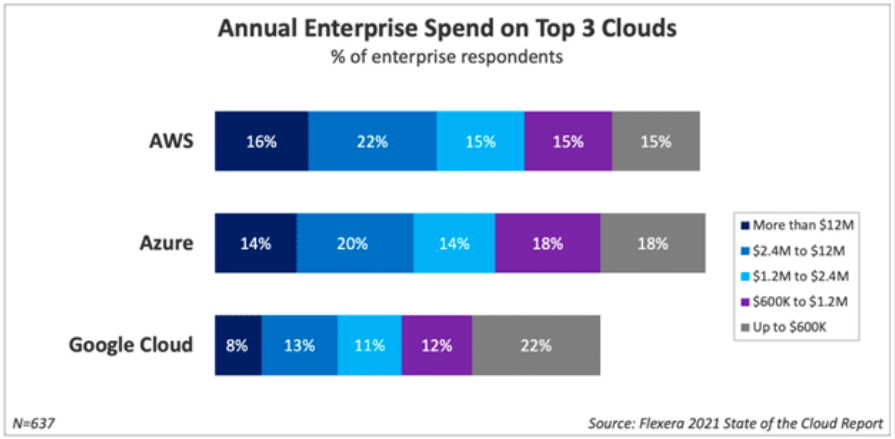


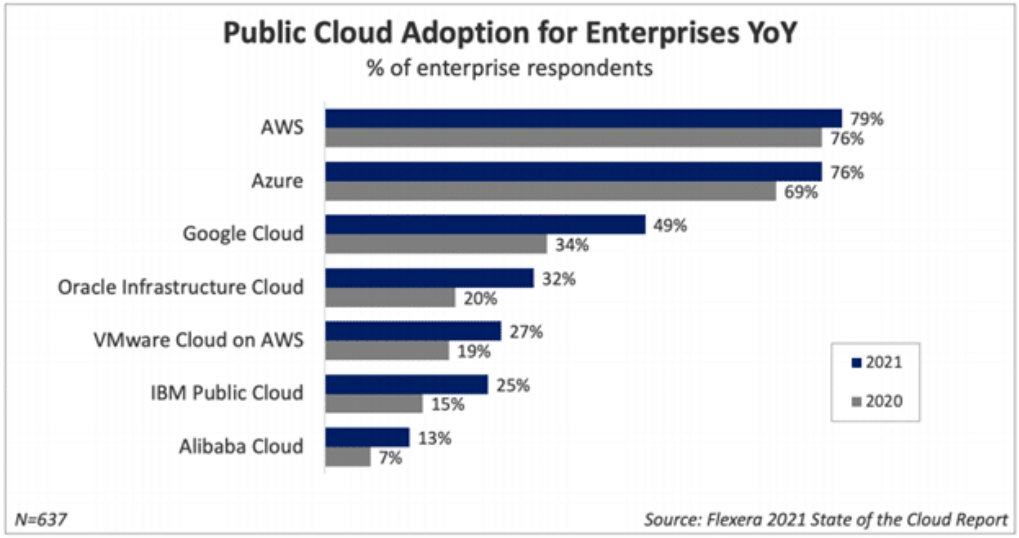


https://www.flexera.com/blog/cloud/cloud-computing-trends-2021-state-of-the-cloud-report/

The State of the Cloud Survey

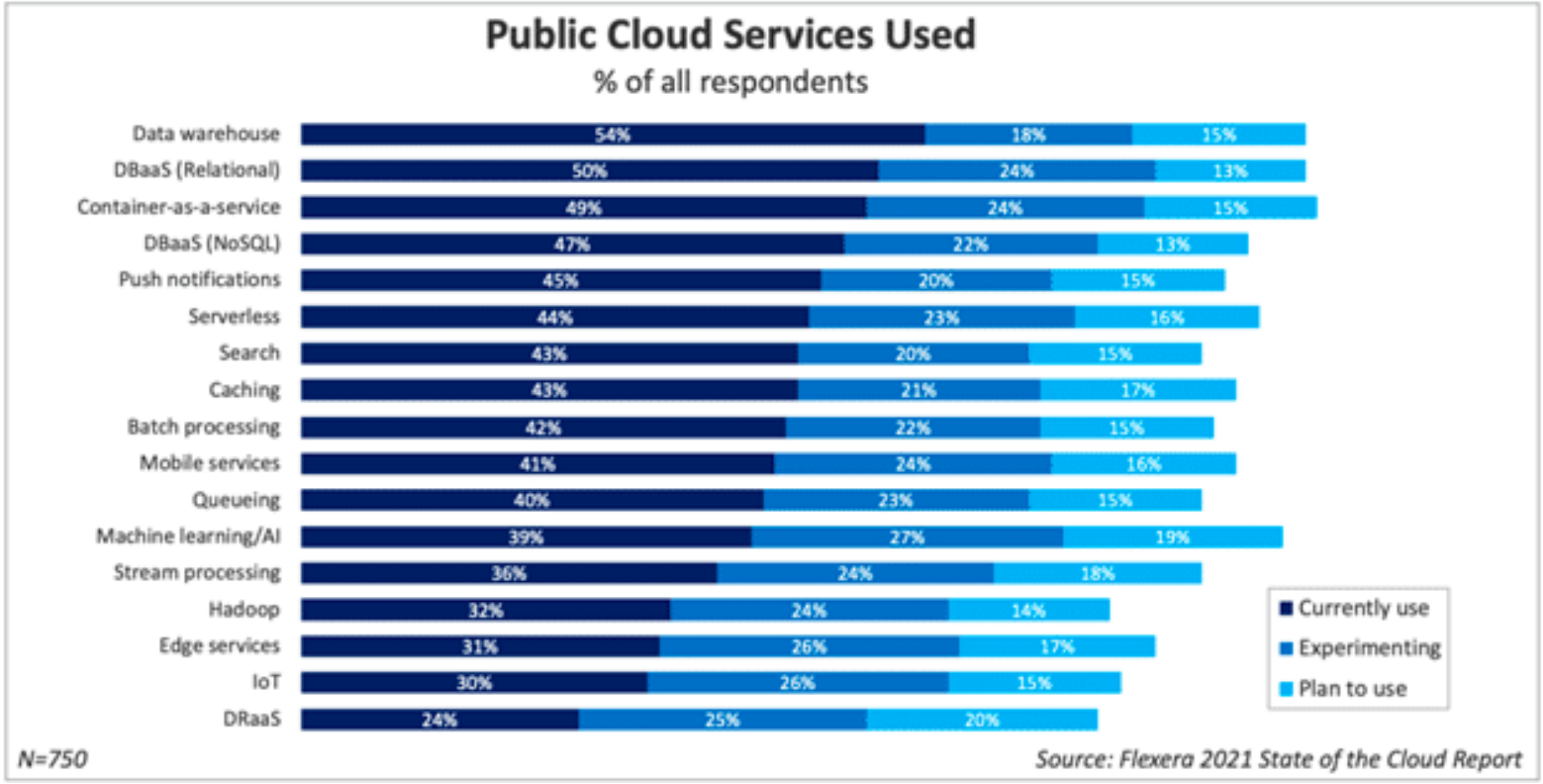






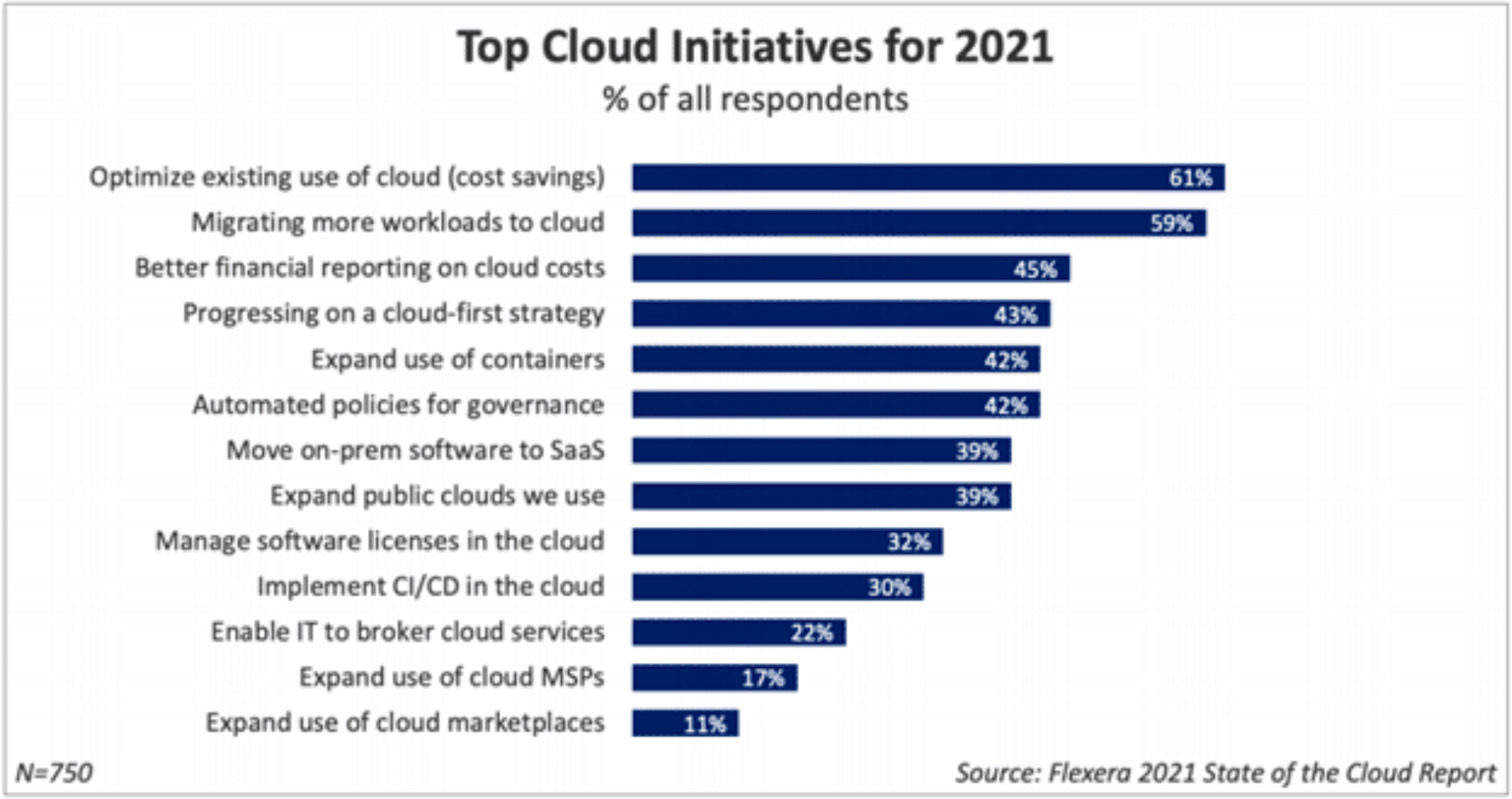
https://www.flexera.com/blog/cloud/cloud-computing-trends-2021-state-of-the-cloud-report/

The State of the Cloud Survey



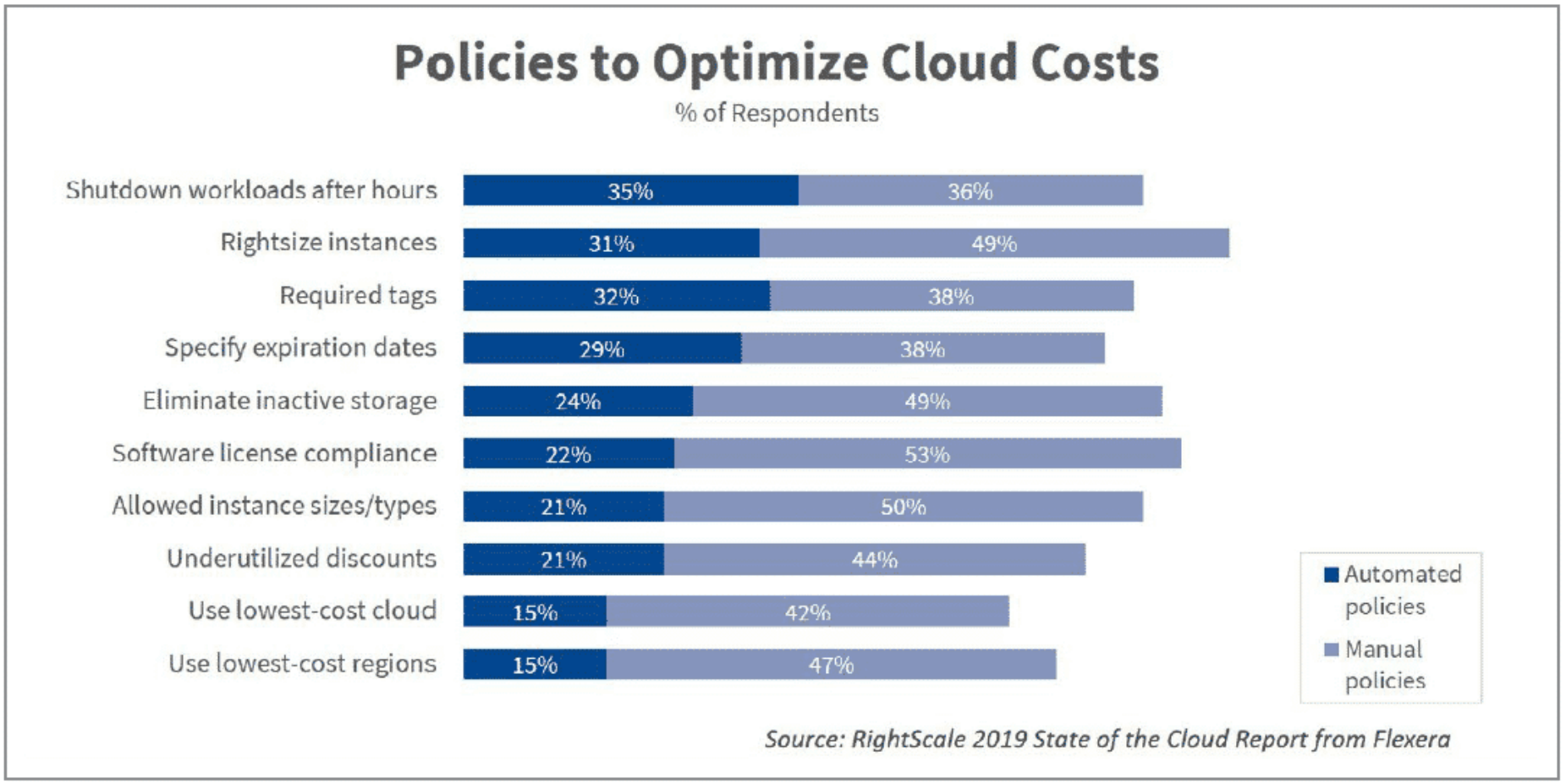
https://www.flexera.com/blog/cloud/cloud-computing-trends-2021-state-of-the-cloud-report/

The State of the Cloud Survey



https://www.flexera.com/blog/cloud/cloud-computing-trends-2021-state-of-the-cloud-report/

The State of the Cloud Survey



https://www.flexera.com/blog/cloud/2019/02/cloud-computing-trends-2019-state-of-the-cloud-survey/

Review Questions

1. What are the 3 main layers of a typical cloud? Give examples of AWS services in each layer. Which ones do your PAs use?
2. What is a benefit of separating PaaS from SaaS in cloud?
3. Briefly explain 1 pro and 1 con of Shared Disk Parallelism vs Shared Nothing Parallelism.
4. Briefly explain 1 pro and 1 con of On-Demand vs Spot instances on AWS.
5. What is so great about the serverless cloud anyway?
6. What is so great about resource disaggregation in future clouds?
7. Briefly explain 2 pros and 2 cons of cloud vs on-premise clusters.