



TOR VERGATA
UNIVERSITÀ DEGLI STUDI DI ROMA

**Macroarea di Ingegneria
Dipartimento di Ingegneria Civile e Ingegneria Informatica**

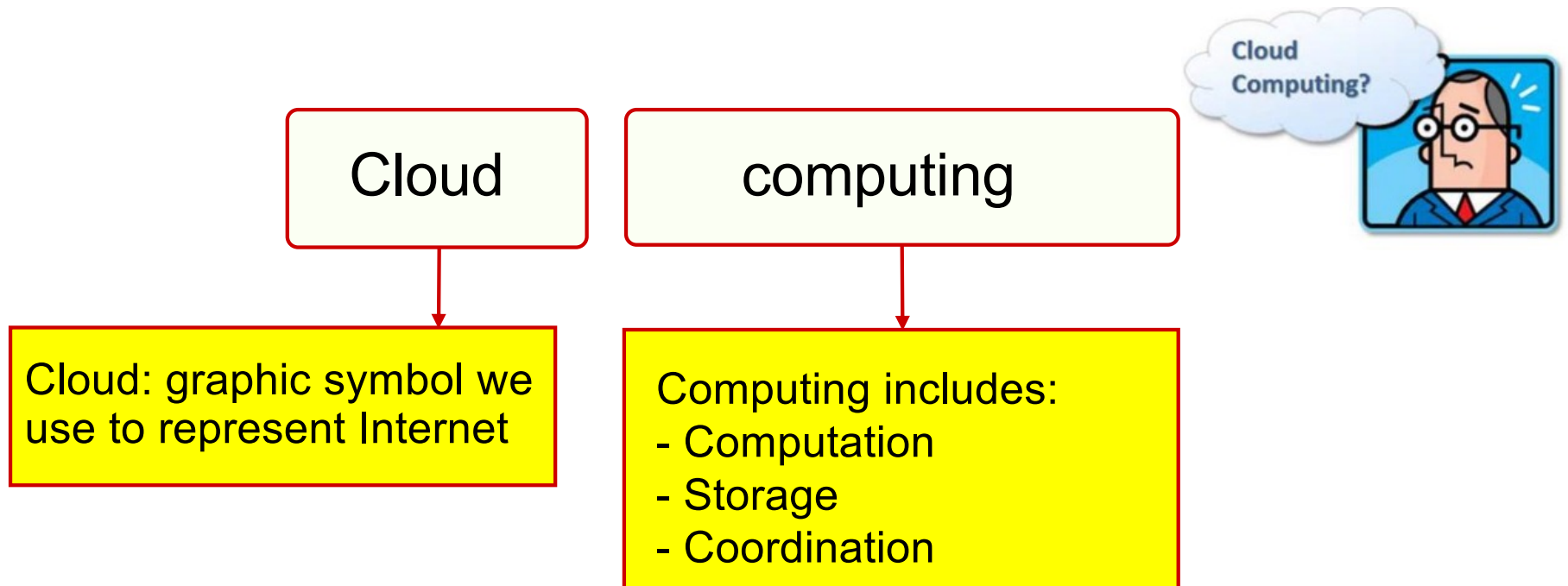
Introduction to Cloud Computing

Corso di Sistemi Distribuiti e Cloud Computing
A.A. 2026/27

Valeria Cardellini

Laurea Magistrale in Ingegneria Informatica

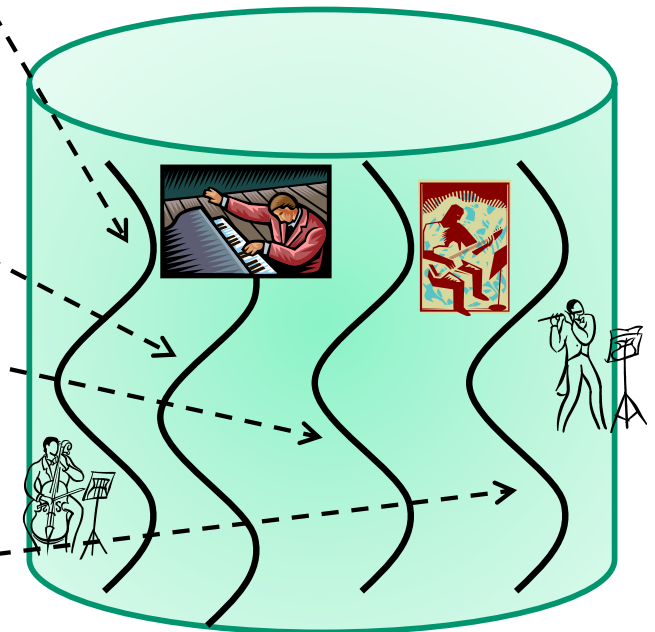
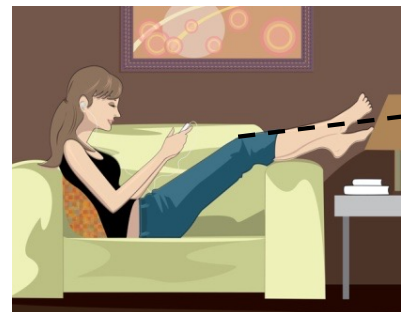
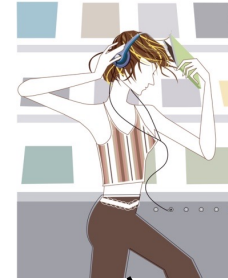
Cloud computing



Cloud computing is about the shift of computing from a single server/data center to Internet

A simple app: classic solution

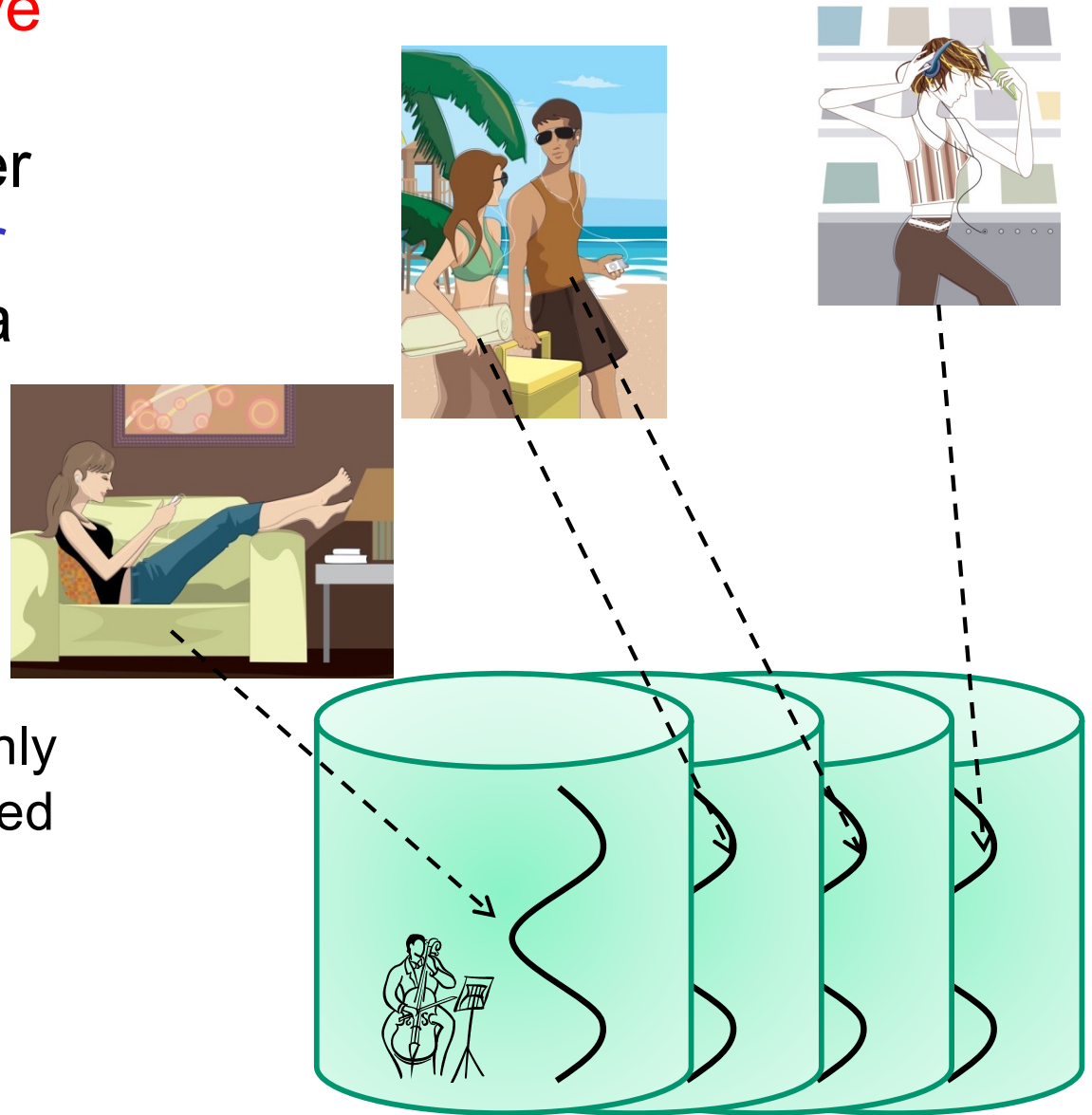
- A simple application: video playback
- How to scale it?
- “**Classic**” solution: **multithreaded** application that exploits multicore parallelism
- Cons:
 - ✗ Complex
 - ✗ Thread crash may impact many users
 - ✗ Scalability limits



A simple app: cloud-native solution

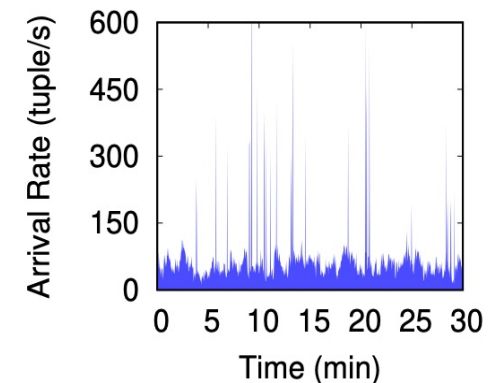
- A simpler **cloud-native** solution: a **single-threaded** video server instantiated **once per user** and running in a **sw container**

- Pros:
 - ✗ Simpler design
 - ✗ If a player crashes, only a single user is affected
 - ✗ Easy to scale out



The real problem: scale and complexity

- How to realize services that:
 - Handle millions of requests per second
 - Manage workload fluctuations of one order of magnitude (or even more) in a quite short period
 - Store EBs of data
 - $1 \text{ EB} = 2^{60} \text{ B} = 10^{18} \text{ B}$



- There is a problem of **scale of services!**
- And scale changes every well-known problem in computer research and industry

Some “old” and partial answers

- Utility computing
- Grid computing
- Autonomic computing
- Software as a Service (**SaaS**)
 - An “old” idea: application delivery on Internet
- ... before cloud computing (2006)
 - One step towards the scale problem solution



The origin: from 4 fundamental utilities...

- Water



- Gas



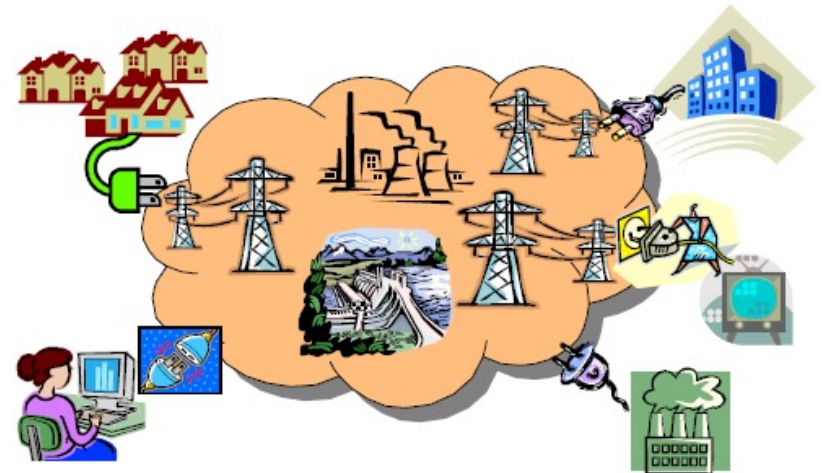
- Electricity



- Telephony/Network



... to **computing** as the
fifth **utility**



Utility computing: new idea?

- But this “computer utility” vision is not new!
- 1961: John McCarthy
 - “If computers of the kind I have advocated become the computers of the future, then **computing may someday be organized as a public utility** just as the telephone system is a public utility... The computer utility could become the basis of a new and important industry.”

John McCarthy

(1927–2011) received the Turing Award in 1971 and was the inventor of Lisp and a pioneer of timesharing large computers. Clusters of commodity hardware and the spread of fast networking have helped make his vision of timeshared “utility computing” a reality.



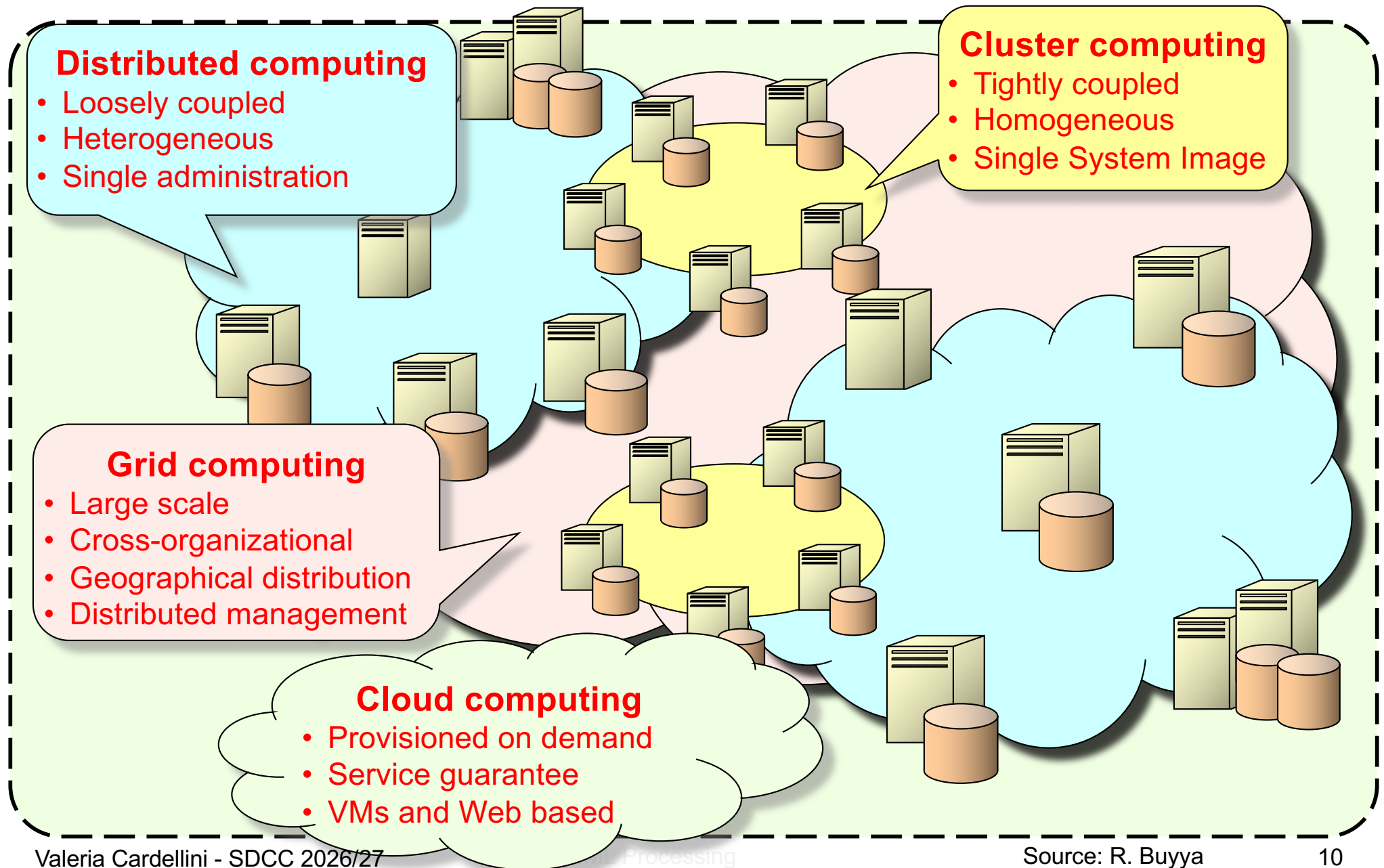
Utility computing: new idea?

- 1969: Leonard Kleinrock, ARPANET project
 - “As of now, computer networks are still in their infancy, but as they grow up and become sophisticated, we will probably see the spread of “**computer utilities**”, which, like present electric and telephone utilities, will service individual homes and offices across the country.”
- Some re-definition of computer
 - 1984: John Gage, Sun Microsystems
 - “**The network is the computer**”
 - 2008: David Patterson, Univ. Berkeley
 - “**The data center is the computer**. There are dramatic differences between of developing software for millions to use as a service versus distributing software for millions to run their PCs”

Towards cloud computing

- 2006: Jeff Bezos, Amazon
 - “Let us use our spare resource for making profit by offering them as services to the public”
 - August 2006: Amazon launched **Elastic Compute Cloud (EC2)** and **Simple Storage Service (S3)**
 - Basic idea: let users rent data storage and computer server time from Amazon like a utility
 - Cloud computing was finally born
- 2011: Rajkumar Buyya, Melbourne Univ.
 - “**Cloud is the computer**”

How do computing paradigms differ?



Many definitions...

- [Armbrust et al., 2010 <https://dl.acm.org/doi/pdf/10.1145/1721654.1721672>]: Cloud computing refers to both the applications delivered as services over the Internet and the hardware and software systems in the data centers that provide those services. The services themselves have long been referred to as Software as a Service (SaaS), so we use that term. The data center hardware and software is what we will call a Cloud. ... Cloud computing has the following characteristics: (1) The illusion of infinite computing resources... (2) The elimination of an up-front commitment by cloud users... (3) The ability to pay for use... as needed.
- [NIST, 2011 <https://csrc.nist.gov/pubs/sp/800/145/final>]: Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.
- [Vaquero et al., 2009 <https://dl.acm.org/doi/pdf/10.1145/1496091.1496100>]: Clouds are a large pool of easily usable and accessible virtualized resources (such as hardware, development platforms and/or services). These resources can be dynamically reconfigured to adjust to a variable load (scale), allowing also for an optimum resource utilization. This pool of resources is typically exploited by a pay-per-use model in which guarantees are offered by the infrastructure provider by means of customized SLAs.

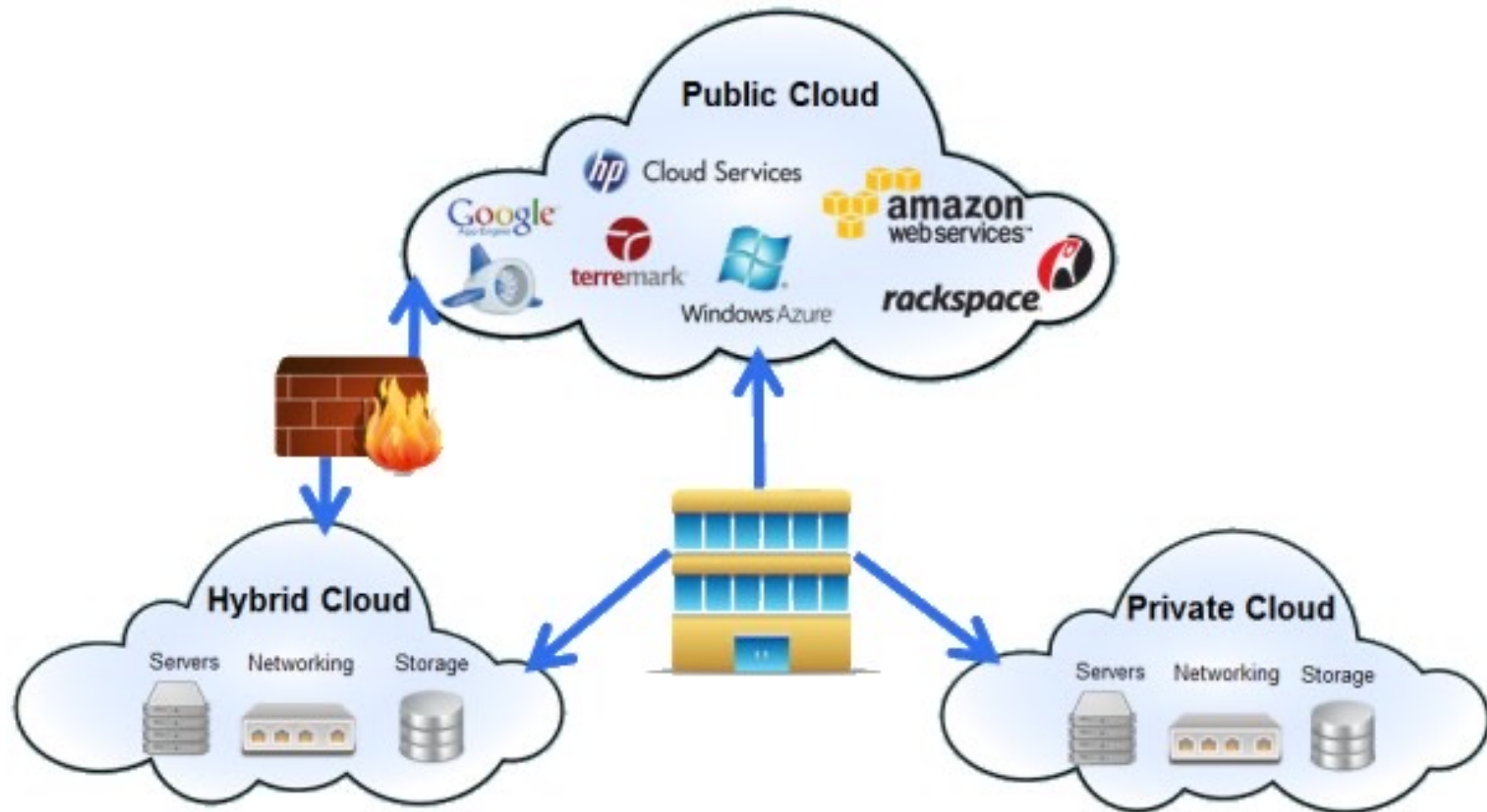
... sharing essential characteristics

- **On-demand self-service**
 - Cloud resources can be provisioned on-demand by users, without requiring interactions with cloud provider
- **Broad network access**
 - Cloud resources accessed over Internet using standard access mechanisms that provide platform-independent access
 - Published service interface/API
- **Elasticity**
 - Ability for customers to quickly request, receive, and later release as many resources as needed
 - Cloud resources can be provisioned rapidly and elastically: they can be rapidly **scaled out/in** based on demand

... that share some essential characteristics

- Resource **pooling**
 - Cloud resources are pooled to serve multiple users using multi-tenancy
 - **Multi-tenancy**: multiple users served by the same physical hardware
- Resource **virtualization**
 - Resources: storage, processing, memory, network bandwidth, and even data centers
- **Pay-per-use** pricing model
 - No large up-front acquisition cost
- **Measured** service
 - Usage of cloud resources is measured and user is charged based on some specific metric

Cloud deployment models



Deployment models: public

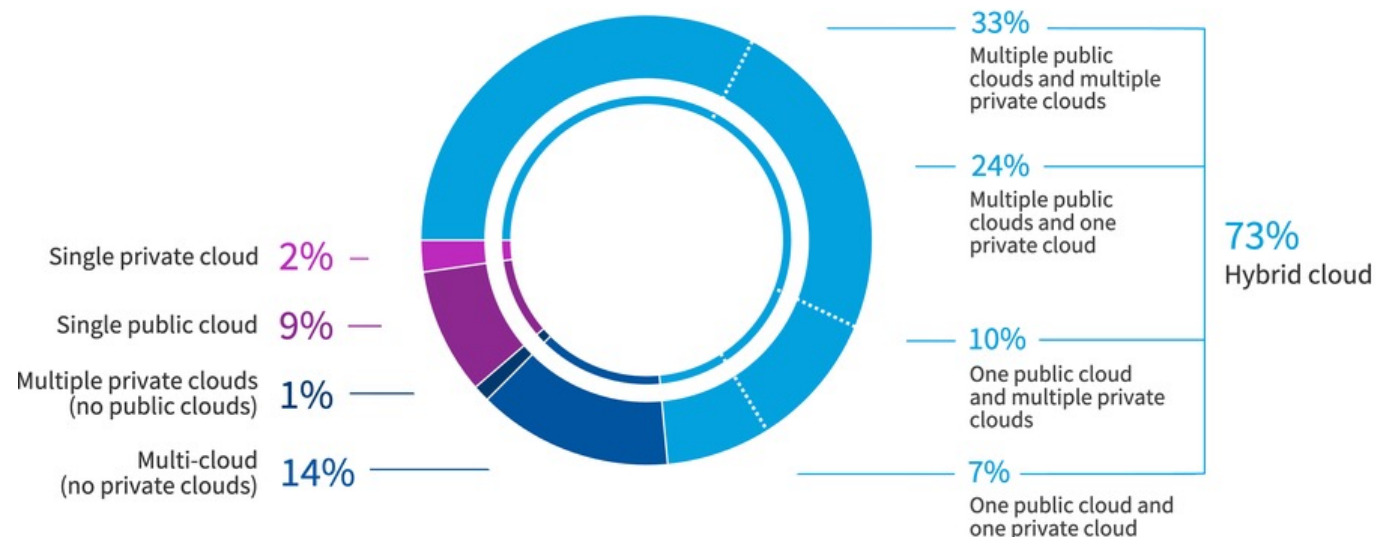
- **Public cloud**
 - Cloud infrastructure: provisioned for **open use** by the **general public** (*multi-tenant*)
 - Management: owned, managed, and operated by a business, academic, or government organization (or a combination)
 - Location: exists **on premises** of cloud provider
 - Cost: services can be free or fee-based

Deployment models: private

- **Private/enterprise cloud**
 - Cloud infrastructure: provisioned for **exclusive** use by a **single organization** (*single-tenant*) comprising multiple consumers (e.g., business units)
 - Management: owned, managed, and operated by the organization, a third party, or some combination of them
 - Location: exists **on-** or **off-premises** of cloud provider
 - ✓ Stronger security, customization
 - ✗ Lower economic advantages, scalability more laborious

Deployment models: hybrid

- **Hybrid cloud**
 - **Mixed** use of private and public clouds
 - Cloud infrastructure: **composition of two or more distinct cloud infrastructures** (private or public) that
 - remain unique entities
 - but are bound together by standardized or proprietary technology that enables **data and application portability**
 - The most popular deployment model



Deployment models: hybrid

- Pros

- ✓ Balance resource utilization and costs

- ✓ Reduce **vendor lock-in** risks

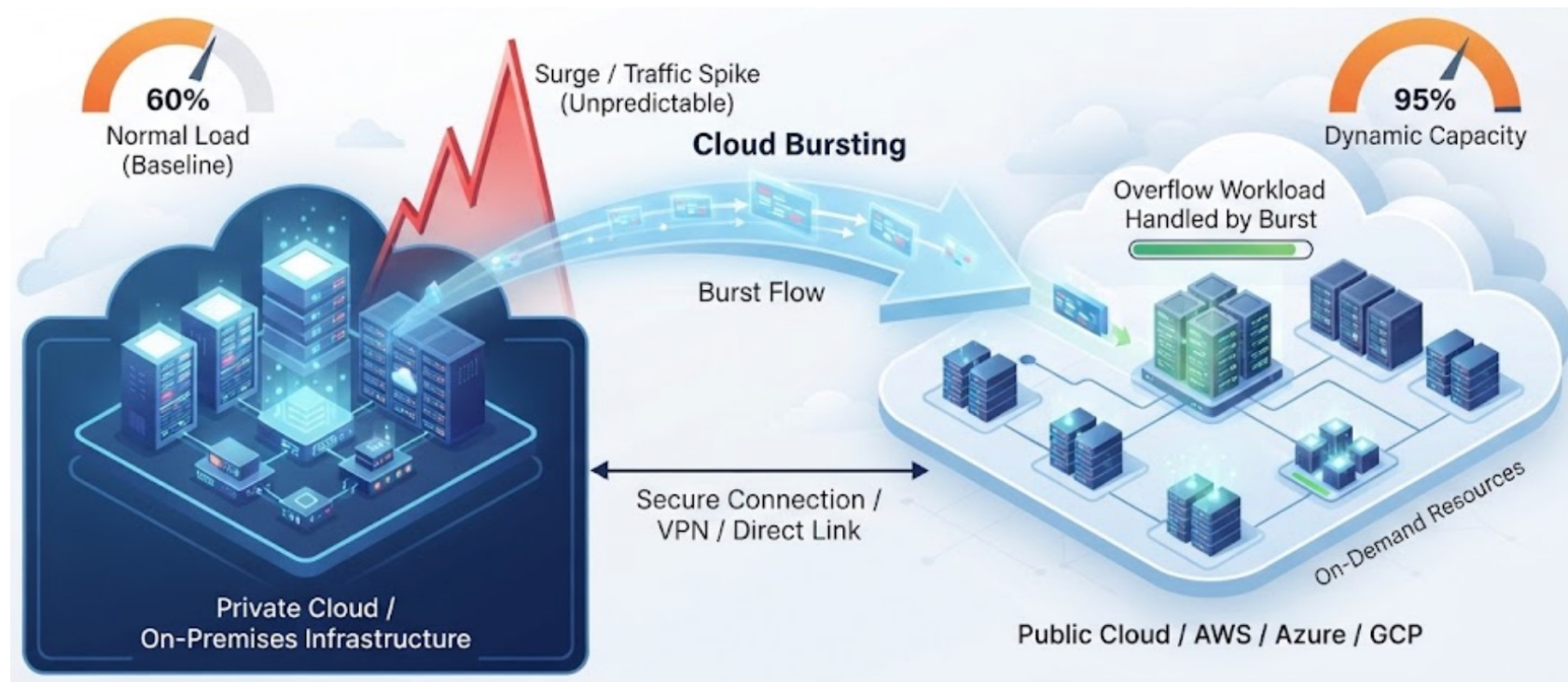
- When migrating away from a provider is too costly or technically difficult, forcing an organization to stay regardless of service quality or price increases

- ✓ Increase flexibility

- Differentiate privacy, e.g., sensitive data in private cloud
 - Differentiate services: apps are siloed on different clouds
 - Improve availability: public cloud for disaster recovery
 - Improve geographic distribution and compliance: public cloud across global regions
 - Improve performance: by means of **cloud bursting**

Hybrid cloud: cloud bursting

- Use public cloud dynamically to handle workload spikes when private cloud capacity is insufficient
 - Private cloud reaches peak capacity
 - Public cloud absorbs spikes



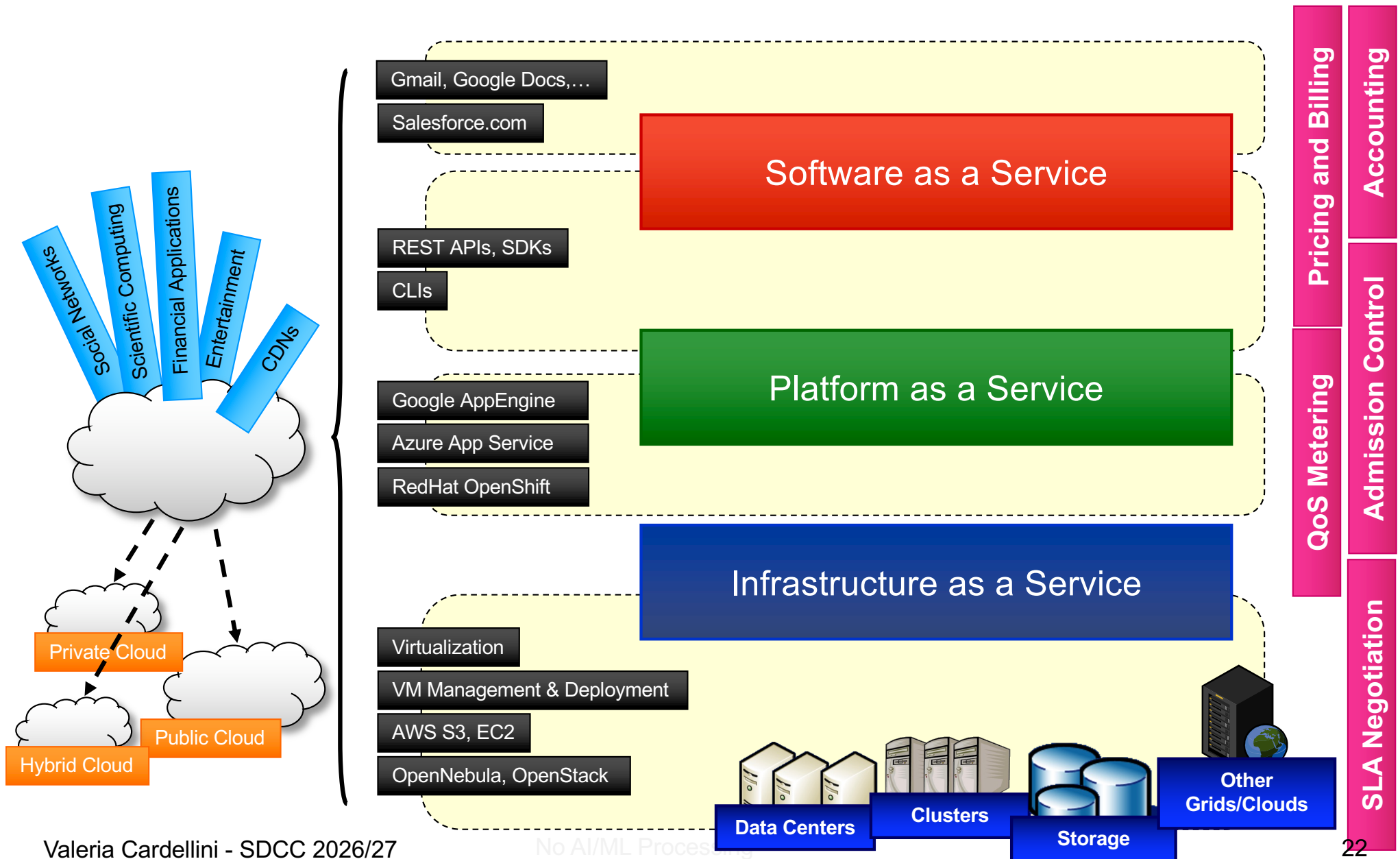
Deployment models: hybrid

- Cons
 - ✗ Operational complexity: managing multiple environments requires specialized tools
 - ✗ Security and compliance: larger attack surface and more complex governance
 - ✗ Performance: cross-cloud data transfer can introduce delays
 - ✗ Troubleshooting: harder across multi-vendor systems
 - ✗ Higher costs: secure connectivity and integration can be expensive

Cloud market and providers

- Cloud providers
 - Amazon Web Services (AWS)
 - Microsoft Azure
 - Google Cloud
 - Alibaba, Cisco, Dell Technologies, DigitalOcean, Huawei, IBM Cloud, Oracle, Rackspace, Salesforce, Tencent... and many more!
- More than 90% of organizations use cloud services
 - Including companies providing popular services
 - E.g., Airbnb, Dropbox, Netflix, Spotify, Zoom
- Cloud market size
 - \$496 billion in 2023
 - Expected to reach \$2000 billion by 2030 (29% IaaS, 30% PaaS, 41% SaaS) <https://bit.ly/4rtpA1C>

Cloud stack: basic service models



Service models: IaaS

- Infrastructure as a Service (IaaS)
 - Compute, storage and network (virtualized) resources as services
 - Customer capability
 - To provision fundamental resources (processing, storage, networks) where consumers deploy and run arbitrary software (including OS and apps)
 - Customer control or management
 - No control over underlying cloud infrastructure
 - Control over OS, storage, deployed apps
 - Limited control of selected network components (e.g., firewalls)

laaS: examples

- Amazon Web Services (AWS): e.g., EC2, S3
- Google Cloud: e.g., Compute Engine, Cloud Storage
- Alibaba Cloud: e.g., Elastic Compute Service
- Aruba Cloud: e.g., Virtual Data Center
- CloudSigma
- DigitalOcean
- IBMCloud : e.g., Cloud Virtual Servers, Cloud Object Storage
- Microsoft Azure: e.g., Virtual Machines, Storage

IaaS: AWS EC2

- Launch virtual machine instance on EC2
- Few steps (2-3 minutes):
 1. Sign in to AWS Management Console
 2. Open Amazon EC2 console
 3. Choose [AWS Region](#) for instance deployment
 4. Choose Launch Instance
 5. Choose software configuration (operating system and other sw) from available [Amazon Machine Images](#) (AMIs)
 - AMI provided by AWS, user community, or AWS Marketplace
 6. Choose resource configuration
 - [T-shirt sizes](#): various combinations of CPU, GPU, memory, storage, and networking capacity
 7. If not yet configured, select security group to open a specific network port (e.g., SSH)
 8. Select an existing key pair (or create a new one) to securely connect to the instance

Service models: PaaS

- Platform as a Service (PaaS)
 - Runtime environment for applications, development and data processing platforms
 - Customers develop, build, run, and scale applications, without building and maintaining the infrastructure
 - Customer capability
 - To develop, deploy, and test apps realized using programming languages, application frameworks, development tools supported by PaaS provider
 - Customer control or management
 - No control over underlying cloud infrastructure (network, servers, OS, storage)
 - Control over deployed apps and possibly application hosting environment configurations

PaaS: examples

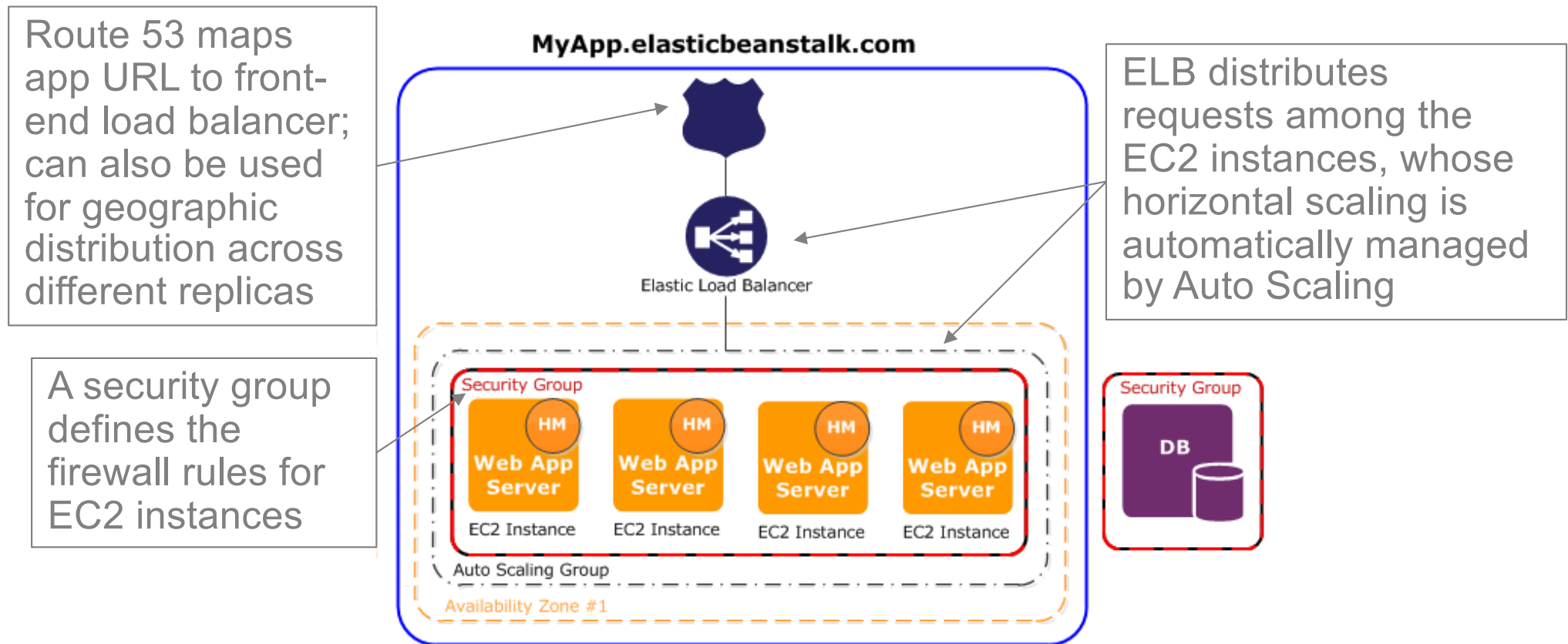
- Platforms for Web apps
 - AWS Elastic Beanstalk, Google App Engine, Azure App Service
- Platforms for building and deploying applications in containers and Kubernetes
 - AWS Elastic Container Service, Google Kubernetes Engine, Azure Kubernetes Service
- Platforms for application lifecycle
 - Tools for CI/CD, container orchestration, and integration with various services
 - Heroku, Red Hat OpenShift, VMware Tanzu
- Salesforce Platform
 - To build, deploy, and manage custom applications within Salesforce ecosystem
- Serverless computing/Function as a Service (FaaS)
 - AWS Lambda, Google Cloud Functions, Alibaba Function Compute

PaaS: AWS Elastic Beanstalk

- Supports applications developed in Go, Java, .NET, Node.js, PHP, Python, and Ruby
- Allows user to deploy and manage applications without having to learn about the infrastructure
- Reduces management complexity
 - User uploads the application source bundle (e.g., .war file) to Elastic Beanstalk and provides some information about the application
 - Elastic Beanstalk automatically launches an environment and creates and configures the AWS resources needed to run the code, handling the details of capacity provisioning, load balancing, scaling, and application health monitoring

PaaS: AWS Elastic Beanstalk

- Example of Elastic Beanstalk architecture
 - AWS resources (ELB, Auto Scaling group, EC2 instances) are automatically provisioned by Elastic Beanstalk



<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/concepts-webserver.html>

IaaS vs. PaaS

- Using IaaS
 - “Pure” virtualized resources: CPU, memory, storage, load balancer, ...
 - OS included
 - “T-shirt” size
- Using PaaS
 - Virtualized resources plus application framework
 - Additional services, e.g., horizontal scaling without having to configure it
 - ✗ Constraints on application and data architecture
 - ✗ Increased risk of vendor lock-in

IaaS and PaaS providers

- Main providers: AWS, Microsoft Azure, Google Cloud, IBM Cloud, Oracle, Alibaba
- Gartner's magic quadrant for strategic cloud platform services (2025)



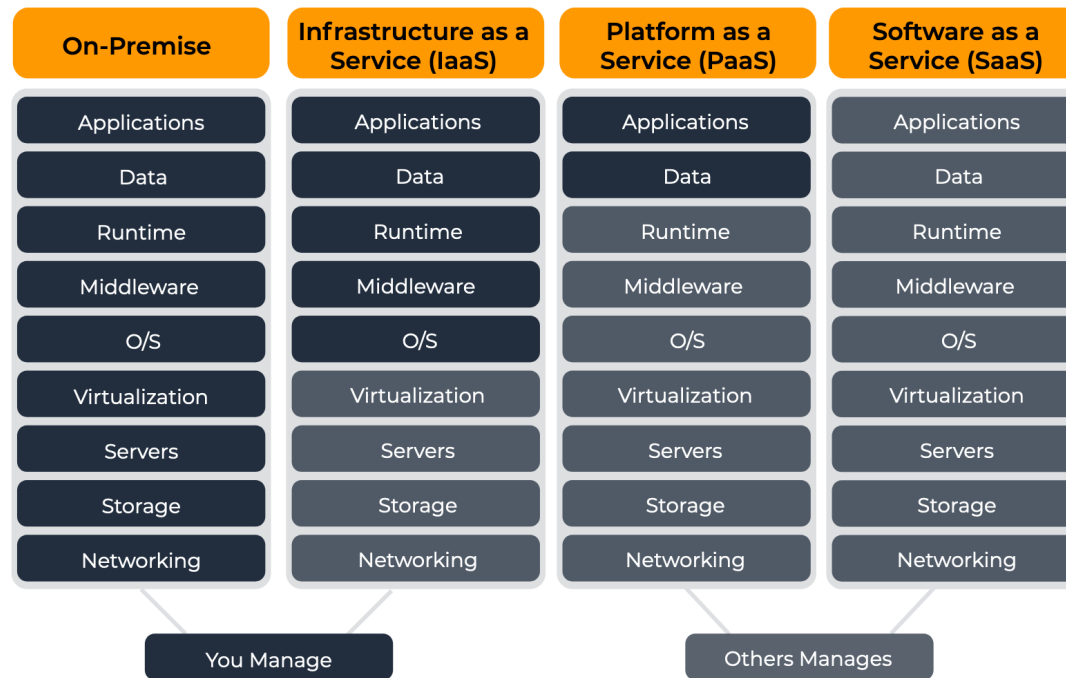
Service models: SaaS

- Software as a Service (SaaS)
 - Applications made available to customers over Internet, still the largest market
 - Customer capability
 - To use SaaS provider's applications running on a cloud infrastructure
 - Applications accessible from various client devices through Web or provider APIs
 - Customer control or management
 - No control over underlying cloud infrastructure
 - Network, servers, operating systems, storage, or even individual application capabilities
 - Possible exception: limited user-specific application configuration settings

SaaS: examples

- Communication & collaboration: Adobe Connect, Cisco WebEx, Google Mail, Zoom
- Personal productivity: Google Calendar, Google Drive, Microsoft 365
- File storage and sharing: Dropbox, OneDrive, iCloud, SugarSync
- CRM (Customer Relationship Management): salesforce.com
- Enterprise-level (accounting, ERP, HR, marketing, sales): NetSuite, SAP Business ByDesign, Workday, Zoho

Service models: wrapping up



- More service models:
 - Database as a Service (DBaaS)
 - Function as a Service (FaaS)
 - Monitoring as a Service (MaaS)
 - Generative AI as a Service (GenAIaaS)
 - ... Anything as a Service (XaaS)

Cloud pricing models

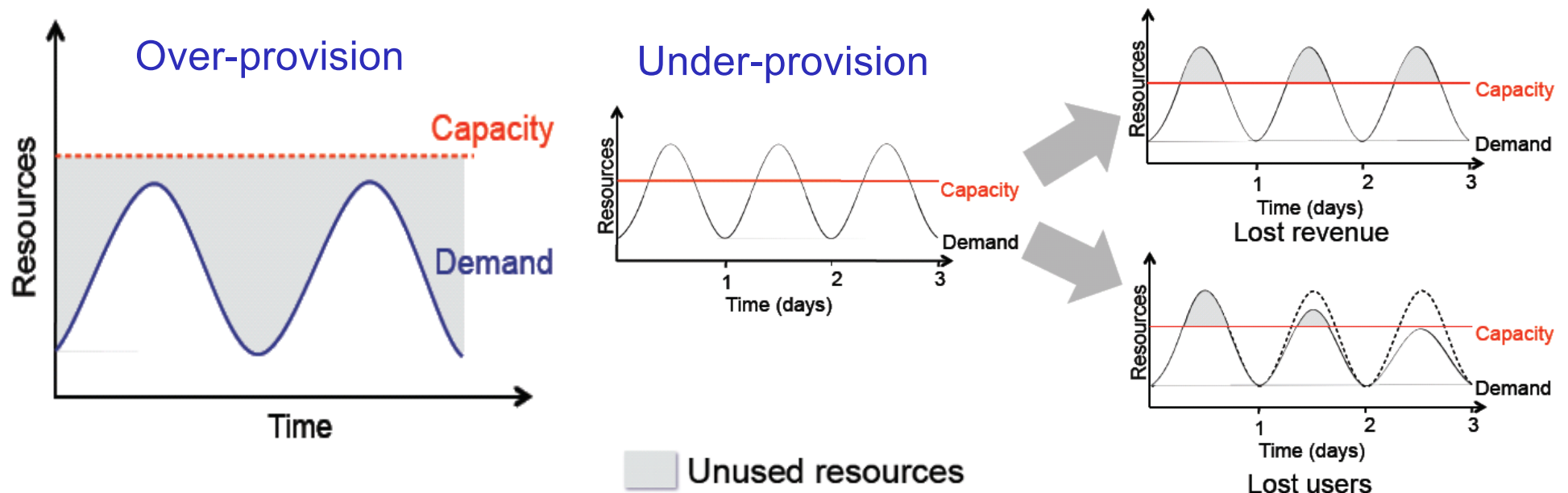
- Cloud providers offer a number of pricing models
- **Pay-per-use / on-demand**
 - Users are charged based on cloud resources usage
 - Pricing granularity depends on service, e.g., by hour for AWS EC2, by ms for AWS Lambda
 - ✓ Flexibility: no need to plan in advance
 - Used for unpredictable traffic, elastic scaling, and dev/test environments
- **Fixed / reserved**
 - Users are charged a fixed amount per month for cloud resources
 - ✓ Significant discount compared to on-demand pricing
 - ✗ Upfront payment
 - Used for predictable, baseline workloads

Cloud pricing models

- **Dedicated**
 - Physical resources (e.g., physical server) rather than virtual ones fully dedicated to user
 - ✓ Dedicated hardware, maximum isolation
 - ✗ Higher cost
- **Spot**
 - Variable pricing for cloud resources driven by market demand (supply and demand)
 - ✓ Substantial savings
 - E.g., AWS EC2 spot instances can be acquired at up a 90% discount compared to on-demand pricing
<https://aws.amazon.com/ec2/spot/>
 - ✗ Interruptible by cloud provider
 - 👉 use for fault-tolerant workloads

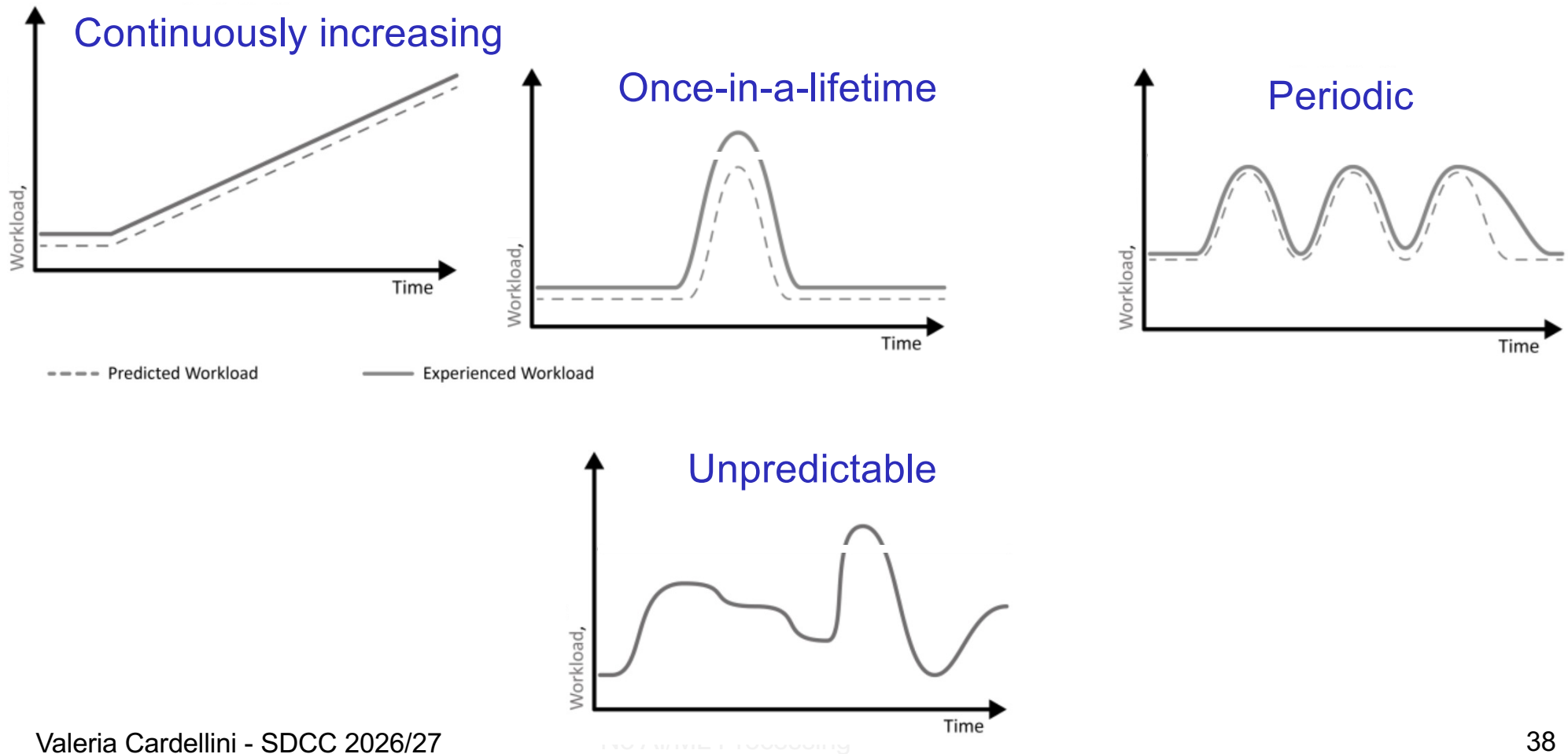
Capacity planning and resource provisioning

- Capacity planning: determines right sizing (of each tier) of application deployment in terms of:
 - Number of IT resources and their size
 - IT resources: computing, storage, memory and network
- **Static provisioning:** capacity is pre-provisioned
 - ✗ Excess capacity (**over-provision** based on peak load) or capacity lack (**under-provision**)



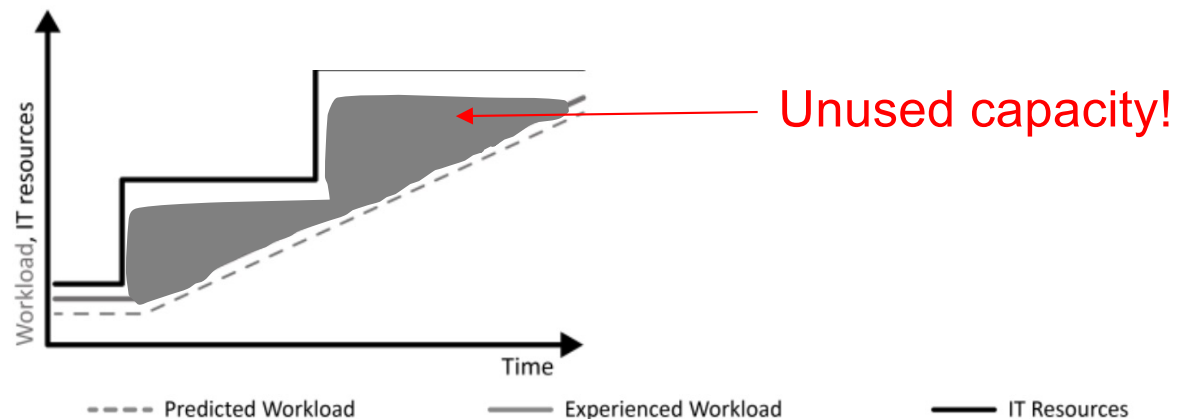
Uncertainty complicates things

- Multiple sources of uncertainty: workload demands, dynamic resource performance due to interference, ...
- Patterns of fluctuating workloads



The classic solution

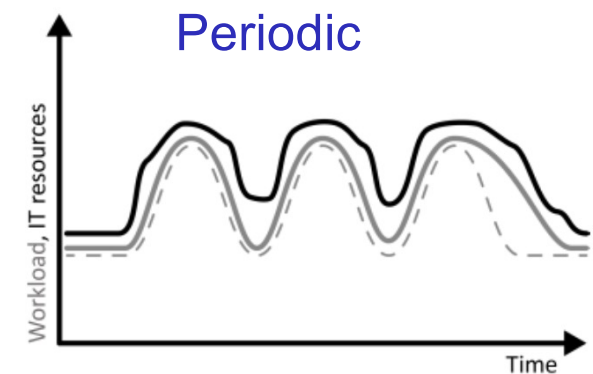
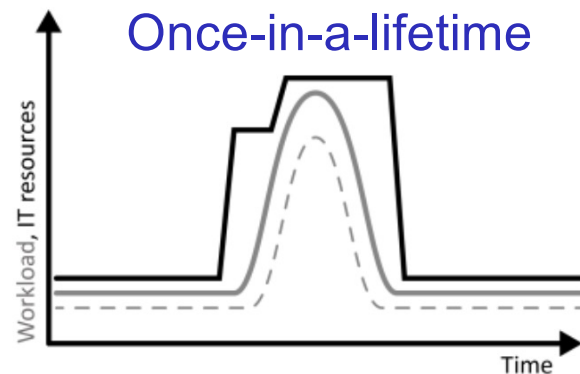
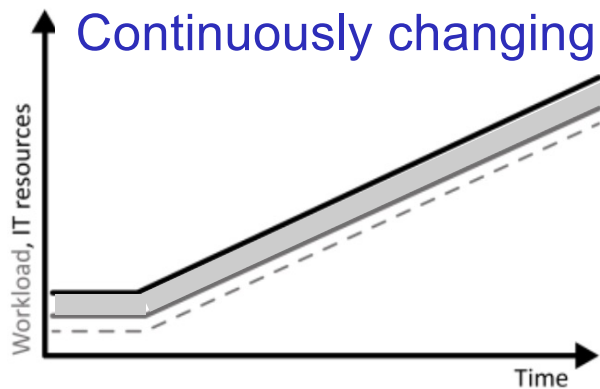
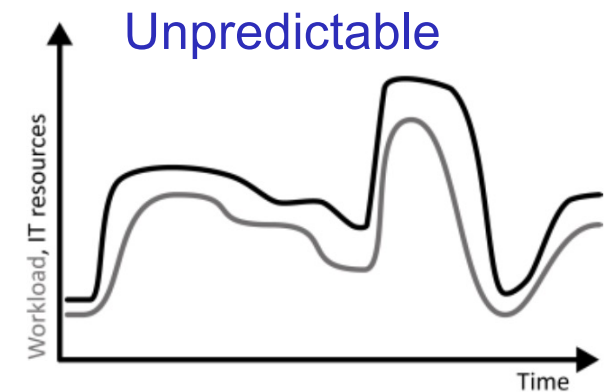
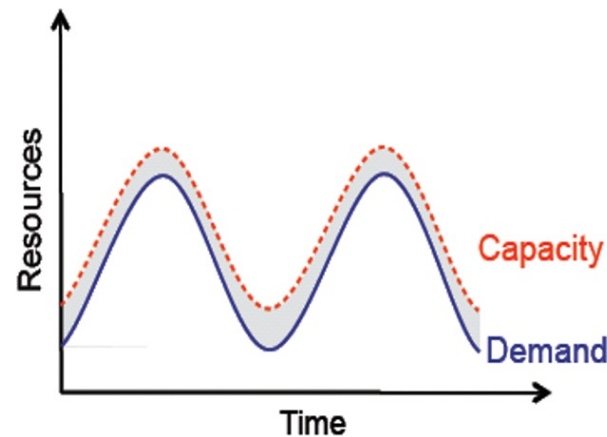
- Forecast demand
- And **manually provision resources dynamically** by scaling in/out (or up/down) IT resources as needed



- Manual scaling is unpractical
 - Takes time, prone to human errors, inefficient, ...
- 👉 Automated scaling is better!

The modern solution: elastic scaling

- Elastic scaling (or **auto-scaling**): **automatically** align IT resources to workloads



--- Predicted Workload

— Experienced Workload

— IT Resources

Elasticity

- (Physics) Property of returning to an initial form or state following deformation

- Stretch when a force stresses it
- Shrink when the stress is removed

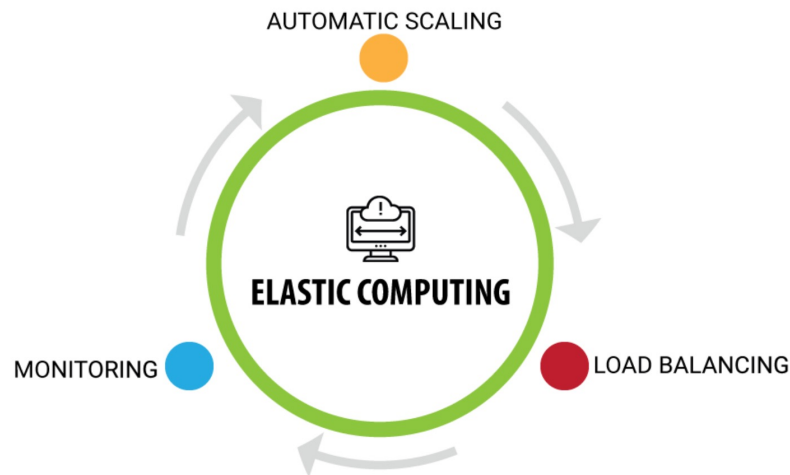


- (Computer science) Degree to which a computing system is able to **adapt to workload changes** by provisioning and de-provisioning resources in an **autonomic** manner, such that at each point in time the **available resources match the current demand as closely as possible**

Herbst et al., Elasticity in Cloud Computing: What It Is, and What It Is Not, ICAC 2013

Elasticity: how to achieve

- Key feature of Cloud computing
- How do we achieve elastic computing?
Through **self-adaptive software systems**



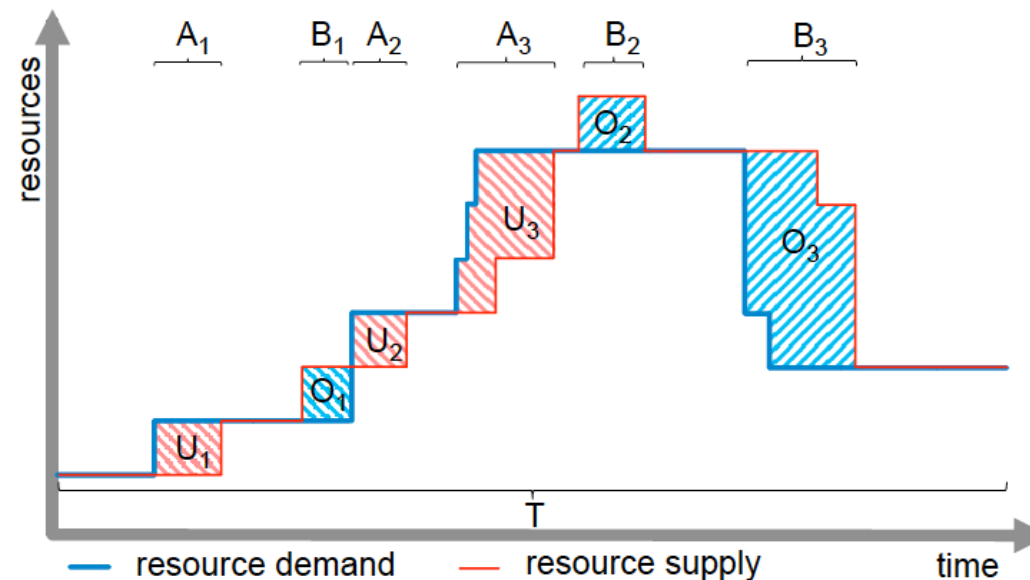
*In any system of sufficiently large scale, **automation** is not only necessary, but it is cheap.*

Ousterhout, Is scale your enemy, or is scale your friend?:
technical perspective, Comm. ACM, 2011

<https://dl.acm.org/doi/pdf/10.1145/1965724.1965748>

Elasticity: how to measure

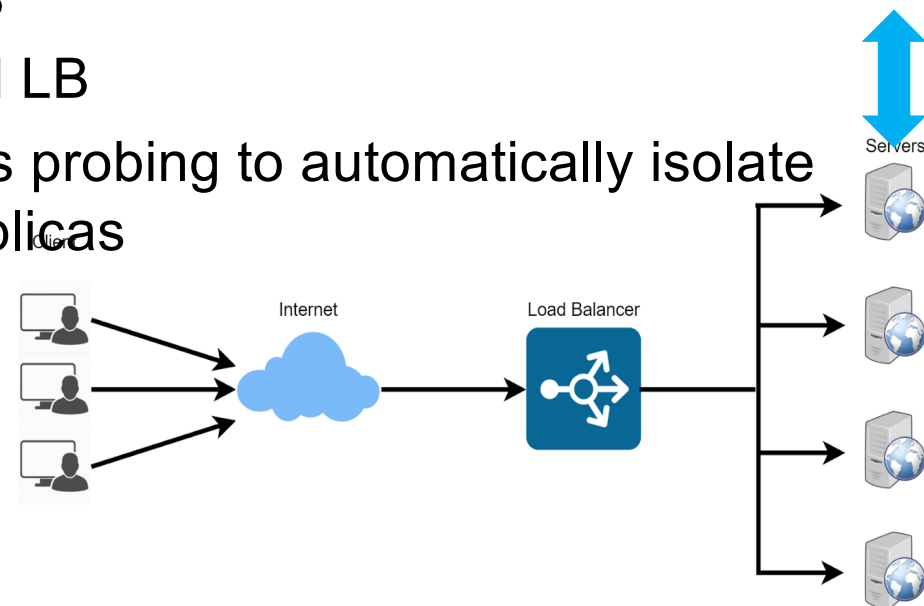
- Consider over-provision and under-provision



- Elasticity metrics:
 - Accuracy**: sum of areas of over-provision (O) and under-provision (U) over the measurement period T
 - Timing**: total amount of time spent in over-provision (B) and under-provision (A) over the measurement period T

Elasticity: architectural point of view

- Distribute requests among multiple server replicas using a *server-side* **load balancer** (LB)
 - A single virtual entry point (VIP) assigned to LB
 - **Network** (layer 4) vs. **application** (layer 7) LB
 - **One-way** LB (direct server response) vs. **two-way** LB (response routed back through LB)
 - **Hardware** vs. **software** LB
 - **Centralized** vs. distributed LB
 - **Health checks**: continuous probing to automatically isolate and remove unhealthy replicas



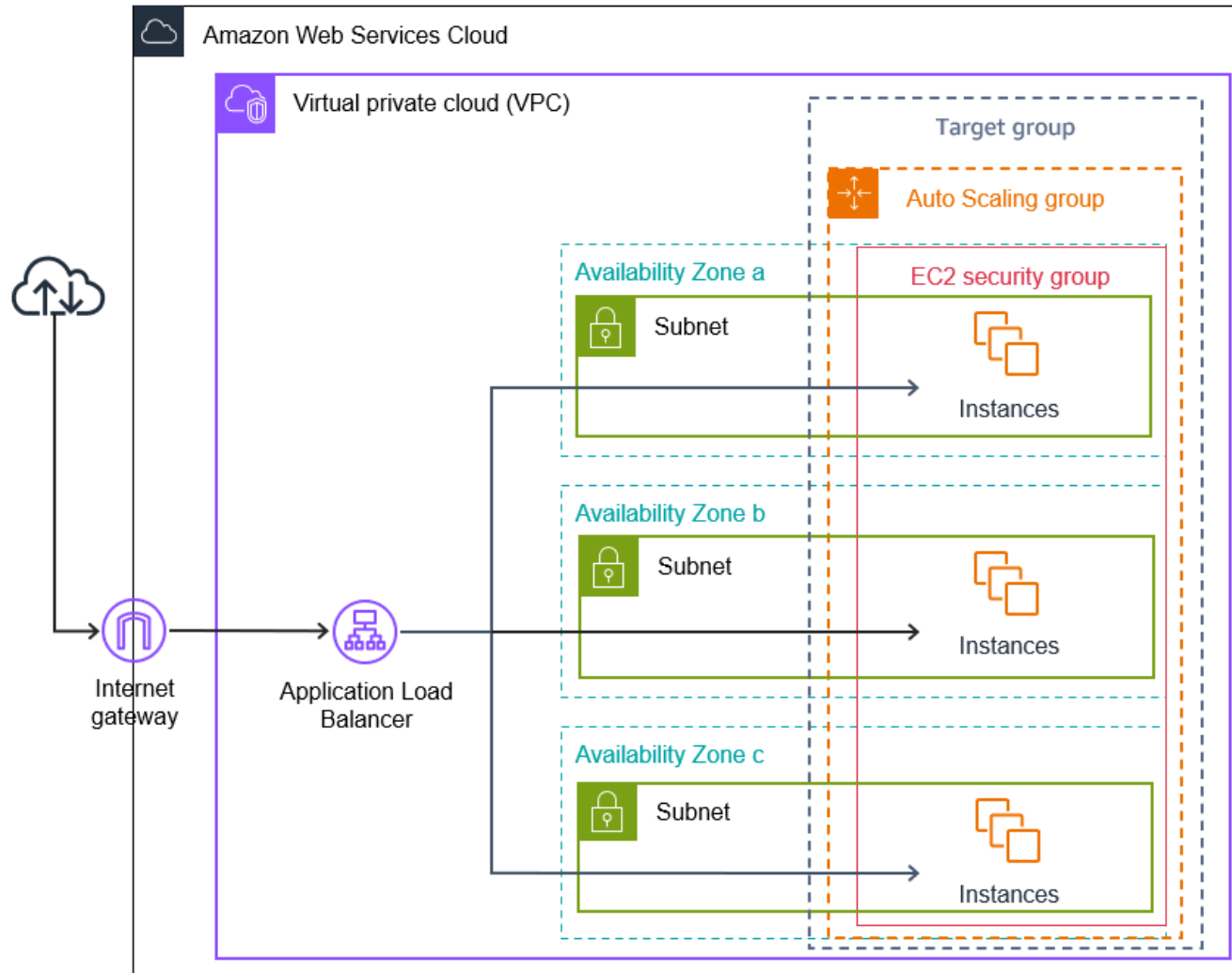
Cardellini et al., The state of the art in locally distributed web-server systems, ACM Comput. Surv., 2002

Elasticity: architectural point of view

- **Load balancing goals**
 - Maximize resource utilization
 - Minimize response time
 - Maximize throughput
- **Load balancing policies**
 - **Stateless**: Random, Round Robin
 - **Server-aware**: Least Loaded, Weighted Round Robin (assign weights proportional to server capacity and distribute requests in a circular way according to weights), Power of Two Choices (select randomly two servers and choose the least loaded of the two)
 - **Client-aware** (depends on layer): Hashing, Session Stickiness
 - **Client & server-aware**: combination of client and server info
- Some products: Envoy <https://www.envoyproxy.io>, nginx https://nginx.org/en/docs/http/load_balancing.html

Elasticity: architectural point of view

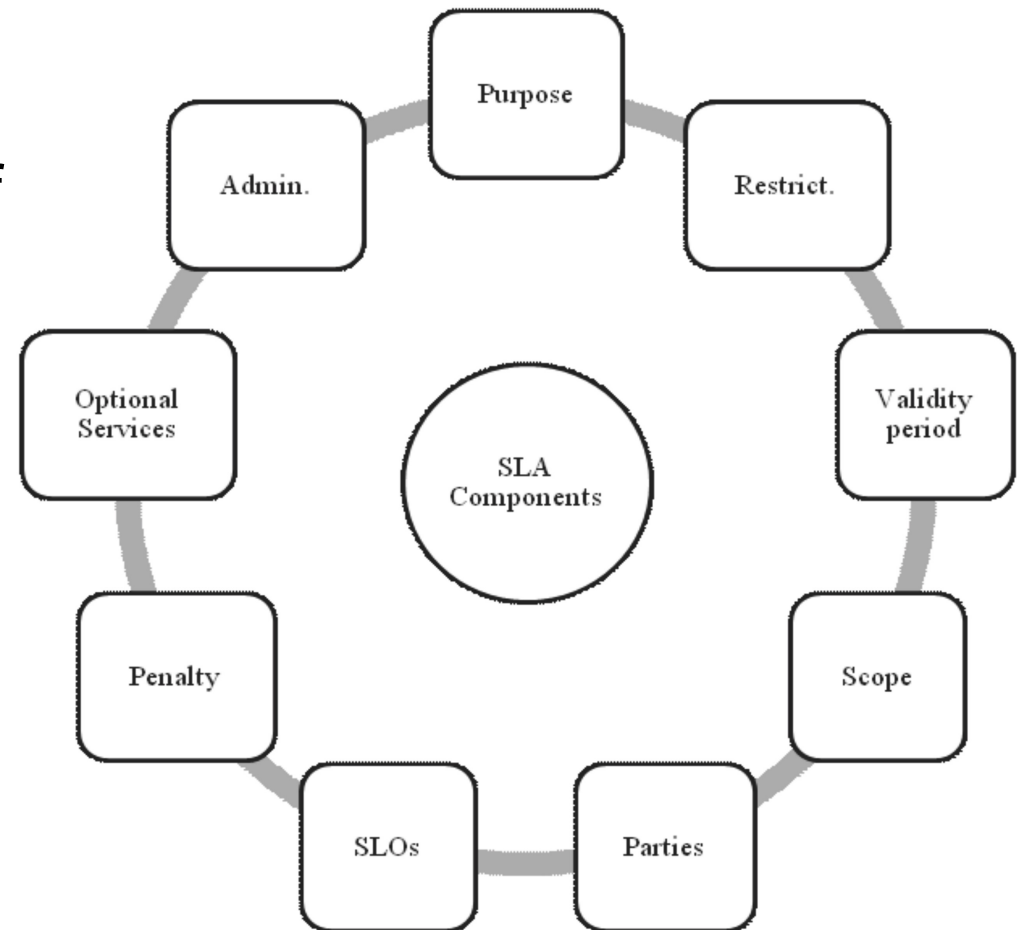
- Example of load balancing and auto-scaling using AWS



<https://docs.aws.amazon.com/autoscaling/ec2/userguide/tutorial-ec2-auto-scaling-load-balancer.html>

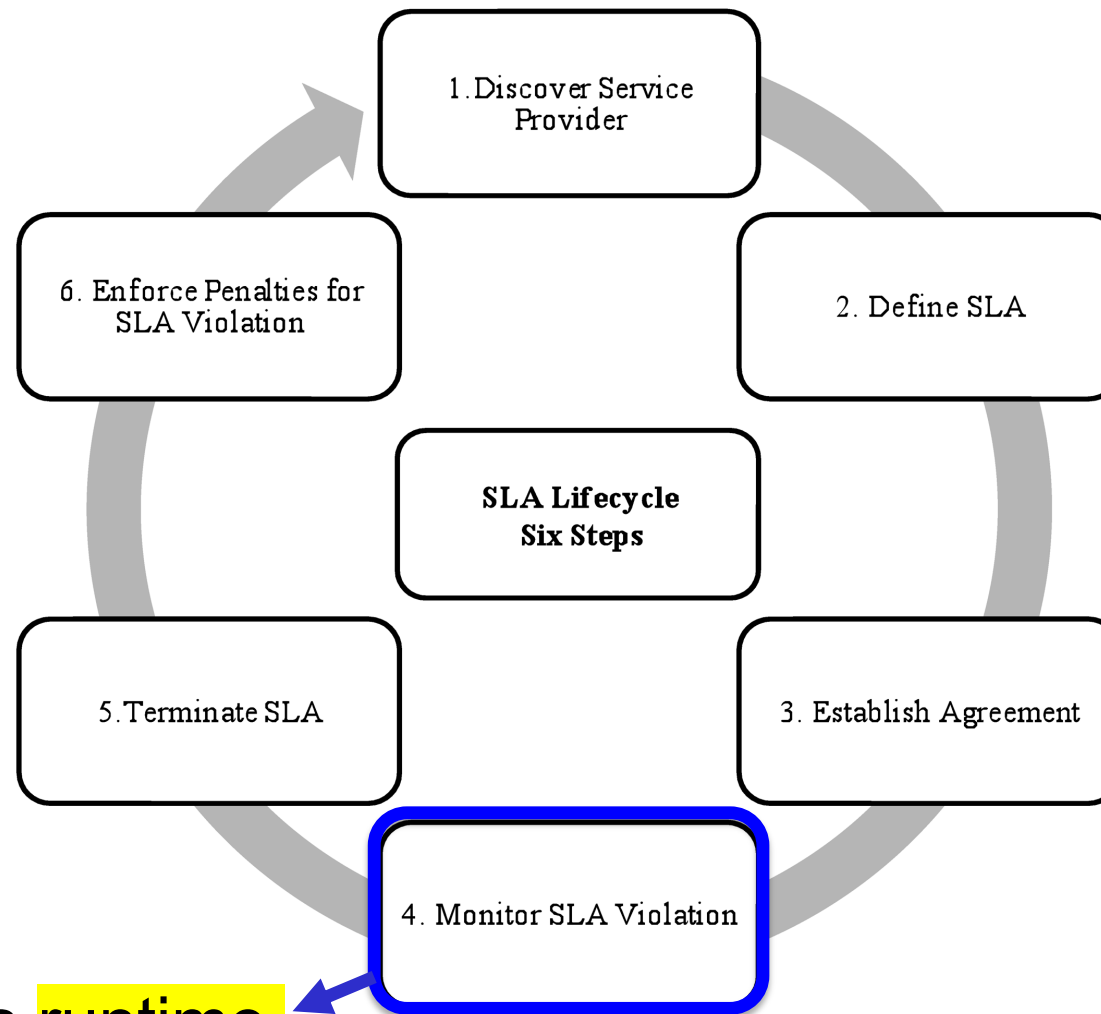
Service Level Agreement (SLA)

- Formal agreement (contract) between a provider and a consumer of a service
- Composed of one or more **Service Level Objectives (SLOs)**:
 - SLO: condition on a measure of a specific metric (e.g., mean response time ≤ 1 s)
- Includes penalty and/or compensation in case of SLA violation



Source: Sun Microsystems Internet Data Center Group

SLA life cycle



Requires runtime monitoring of service

SLI and SLO

- Service Level Indicator (SLI)
 - A measureable metric
 - Examples: availability, response time, throughput
 - Availability is the most common SLI for cloud services
- Service Level Objective (SLO)
 - A predicate over a set of SLIs
 - Example: a fixed threshold on a single SLI
- Example:

monthly uptime percentage for a VM is at least 99.99%

↑
SLI

↑
SLO

Example: Amazon Compute SLA

- **Service Commitment**

https://aws.amazon.com/compute/sla/?nc1=h_ls

For each individual Amazon EC2 instance (“Single EC2 Instance”), AWS will use commercially reasonable efforts to make the Single EC2 Instance available with an Instance-Level **Uptime Percentage of at least 99.5%**, in each case during any **monthly billing cycle** (the “Instance-Level SLA”). In the event any Single EC2 Instance does not meet the Instance-Level SLA, you will be **eligible to receive a Service Credit**

Instance-Level Uptime Percentage	Service Credit Percentage
Less than 99.5% but equal to or greater than 99.0%	10%
Less than 99.0% but equal to or greater than 95.0%	30%
Less than 95.0%	100%

Monthly Uptime Percentage 99.5% = 3h 39m 8s of downtime per month

Monthly Uptime Percentage 99% = 7h 18m 17s of downtime per month

- **Service Credits**

Service Credits are calculated as a percentage of the monthly bill... We will apply any Service Credits **only against future payments**... To receive a Service Credit, **you must submit a claim**...

Example: Amazon S3 SLA

https://aws.amazon.com/s3/sla/?nc1=h_ls

- **Service Commitment**

AWS will use commercially reasonable efforts to make Amazon S3 **available** with a **Monthly Uptime Percentage** during any **monthly billing cycle**. In the event Amazon S3 does not meet the Service Commitment, you will be eligible to receive a Service Credit as described below.

Monthly Uptime Percentage 99.9% = 43 min 49 sec of downtime per month

- **Definitions**

- "Error Rate" means, for each request type to an Amazon S3 storage class: (i) the total number of internal server errors returned by Amazon S3 for such request type to the applicable Amazon S3 storage class as error status "InternalError" or "ServiceUnavailable" divided by (ii) the total number of requests for such request type during the applicable 5-minute interval of the monthly billing cycle. We will calculate the Error Rate for each AWS account as a percentage for each 5-minute interval in the monthly billing cycle. The calculation of the number of internal server errors will not include errors that arise directly or indirectly as a result of any of the Amazon S3 SLA Exclusions.
- "Monthly Uptime Percentage" is **calculated by subtracting from 100% the average of the Error Rates from each 5-minute period in the monthly billing cycle**. If you did not make any requests in a given 5-minute interval, that interval is assumed to have a 0% Error Rate.

Issues with cloud SLAs

- SLA jargon
- Availability
 - Averaging over a period can hide the difference between many short outages vs. a few long ones
- Lack of SLIs for response time
- Service credits: only applicable to future payments
- SLA violation detection: burden of detecting violations often on customers
- Time period for reporting SLA violations
- Non-negotiable (or customizable) SLA for most customers
- Not easy to compare SLAs across different providers

Cloud monitoring

- Goal: track and assess health of Cloud systems and services
 - Ensure that performance issues, outages, or inefficiencies are quickly detected
- Requires tools to **collect** and **analyze** data on **system-oriented and application-oriented metrics** from cloud resources and services

- **System-oriented metrics**

Type	Metrics
CPU	CPU utilization
Disk	Disk utilization, throughput (MB/sec for read/write, operations/sec)
Memory	Memory-Used, Memory-Free, Page-Cache
Interface	Throughput (Mbps incoming/outgoing)

- **Application-oriented metrics**: response time, error rate, transaction volume, user interaction, ...

Cloud monitoring: services and tools

- Examples of monitoring services offered by cloud providers
 - Amazon CloudWatch <https://aws.amazon.com/cloudwatch>
 - Provides monitoring for AWS cloud resources and applications, including system-level and applications metrics
 - Supports alarms, automated responses, and detailed logging for troubleshooting
 - Google Cloud Operations Suite <https://cloud.google.com/monitoring>
- Examples of third-party tools
 - Prometheus <https://prometheus.io>
 - Open-source monitoring tool, collects and stores metrics in a time series DB, which can then be queried for real-time analysis
 - Datadog <https://www.datadoghq.com>
 - Dynatrace <https://www.dynatrace.com>

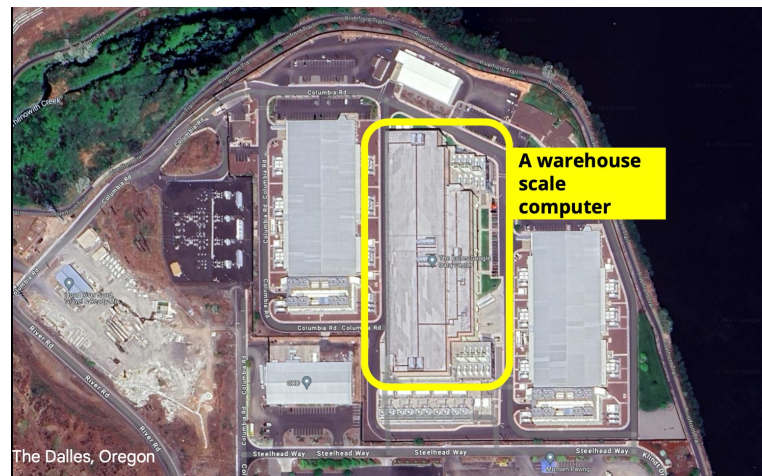
Cloud monitoring: services and tools

- AWS CloudWatch dashboard (example)



Cloud data centers

- Data centers are not only a collection of co-located servers wired-up together
- Cloud services run on **warehouse-scale computers (WSC)**, which are *hyperscale* systems composed of thousands of servers
 - Interconnected servers are viewed as a massive computer

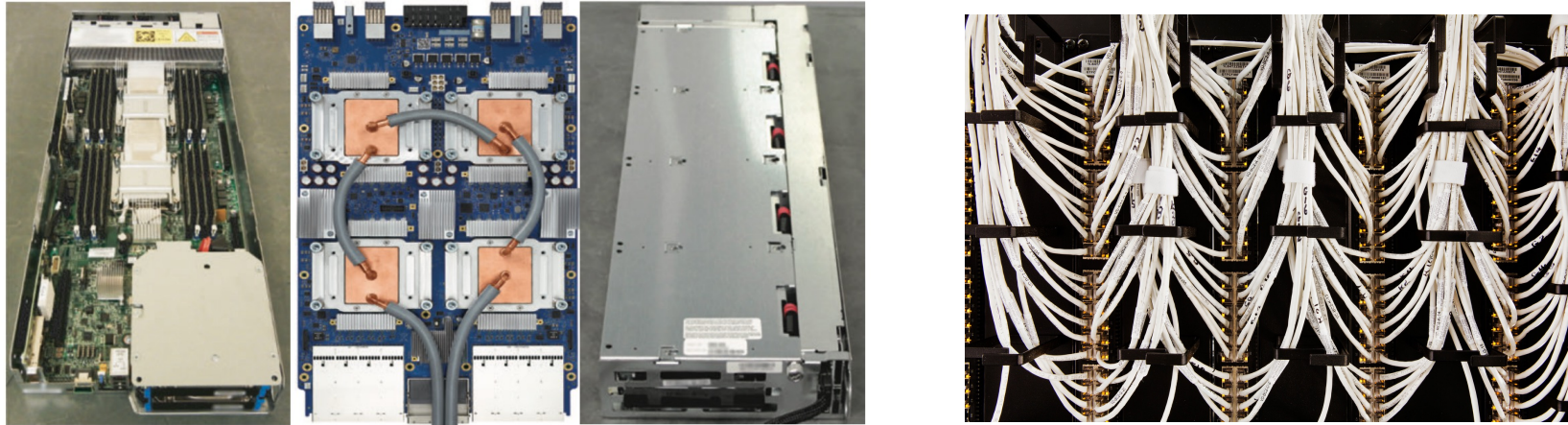


Source: John Wilkes (Google)

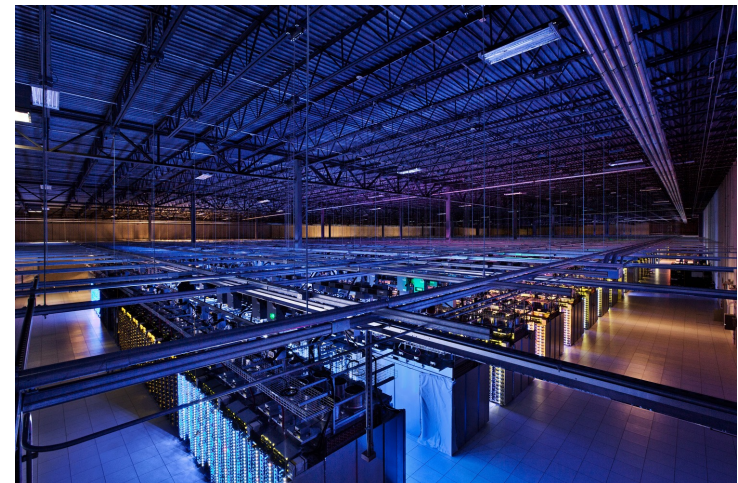
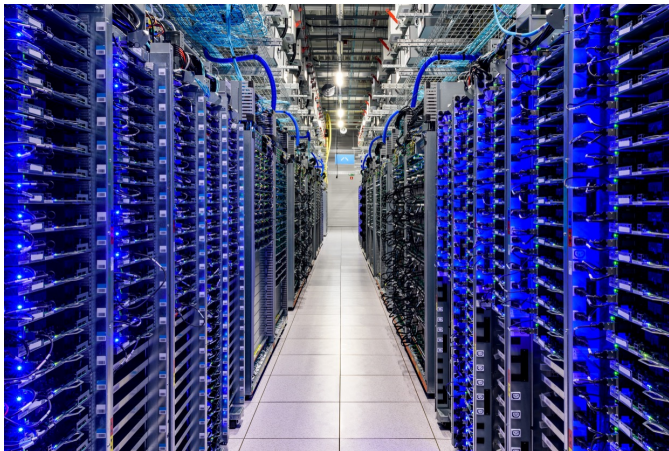
Barroso et al., The Datacenter as a Computer: Designing Warehouse-Scale Machines, 3rd ed., 2019 <https://pages.cs.wisc.edu/~shivaram/cs744-readings/dc-computer-v3.pdf>

Inside a cloud data center

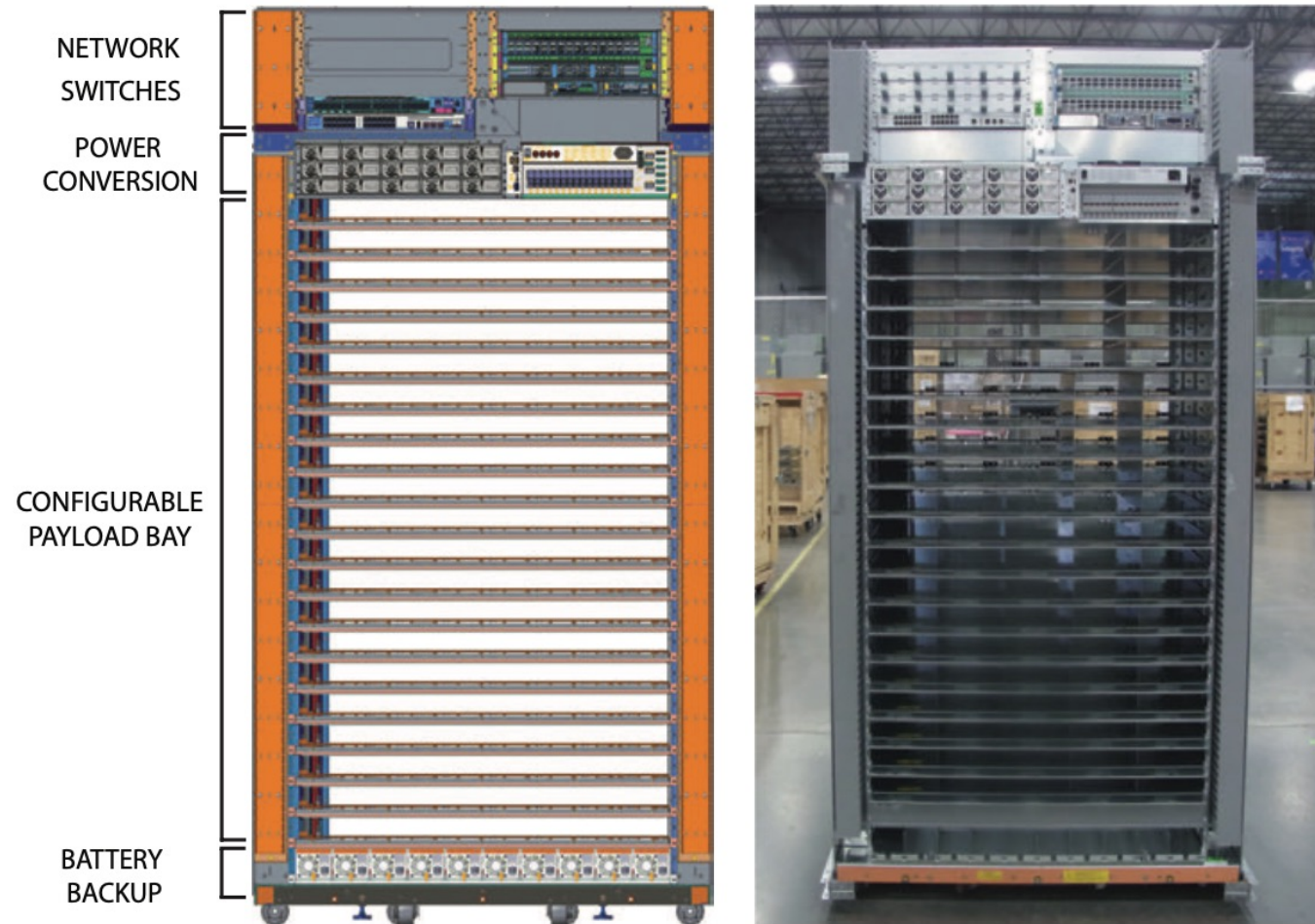
- Hardware building blocks



- ... which are assembled into interconnected racks and rows



Inside a cloud data center: server rack



Outside a cloud data center



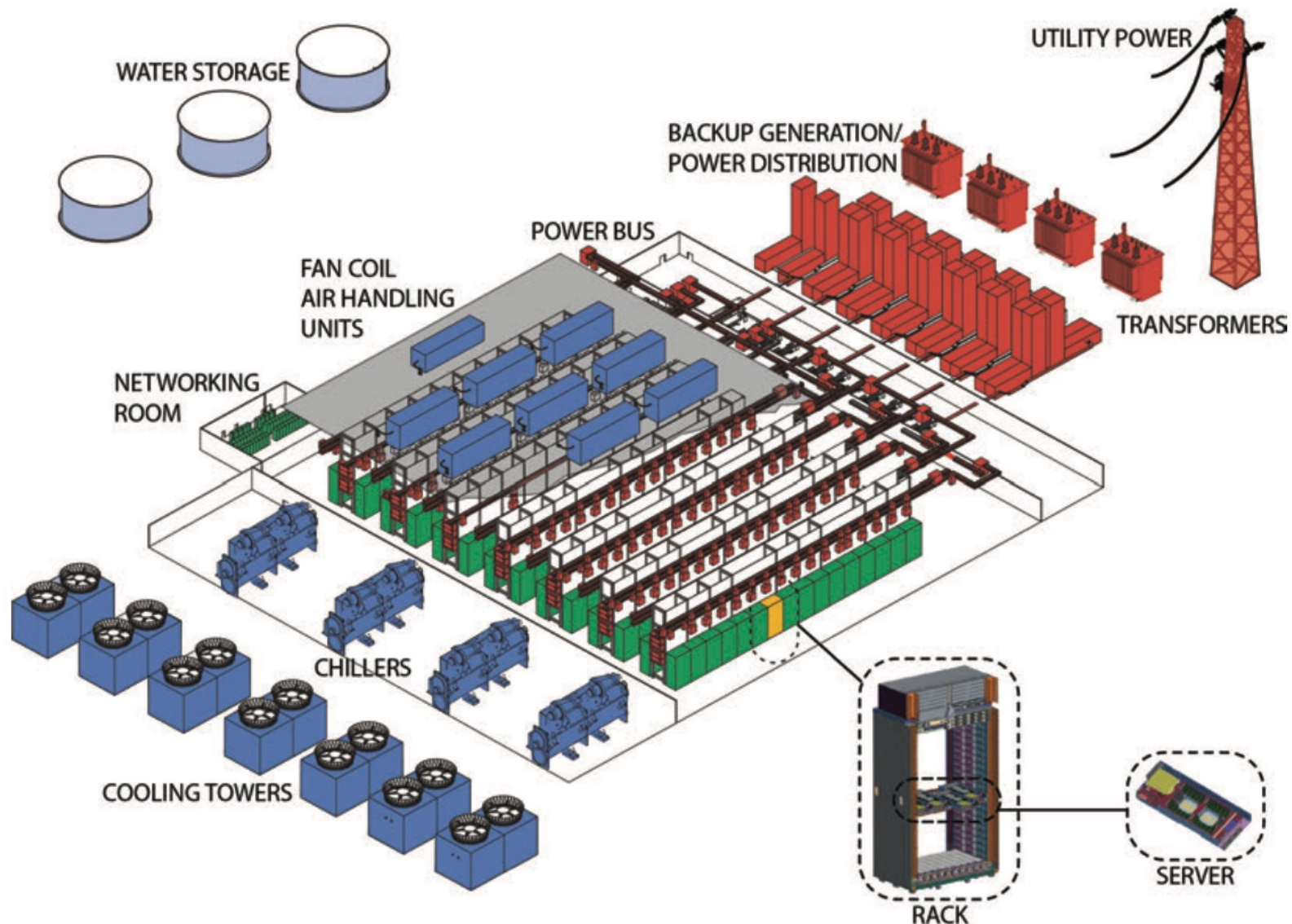
Cooling towers

- How large? $\sim 100000 \text{ m}^2$
- How many servers? $\sim 10^6$
- Power-efficient?

Power Usage Effectiveness (**PUE**) = Facility power / IT equipment power

- E.g., average PUE for Google data centers is 1.1

Main components of a cloud data center



Geographic distribution of cloud data centers

- Where are Cloud data centers?
 - E.g., Amazon Web Services and Google Cloud



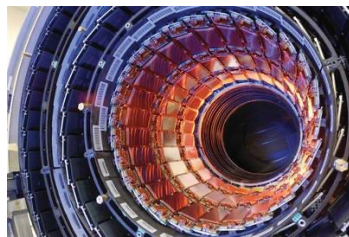
<https://aws.amazon.com/it/about-aws/global-infrastructure> <https://cloud.google.com/about/locations>

- Typically deployed in few locations due to special infrastructure requirements (space, power, cooling, security, technicians, ...)
- Planning and building a new one requires years

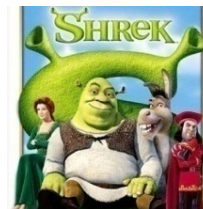
Cloud applications

- Scientific and technical apps
- Healthcare apps
- Consumer and social apps
- Business apps
- ...

Healthcare, scientific and technical

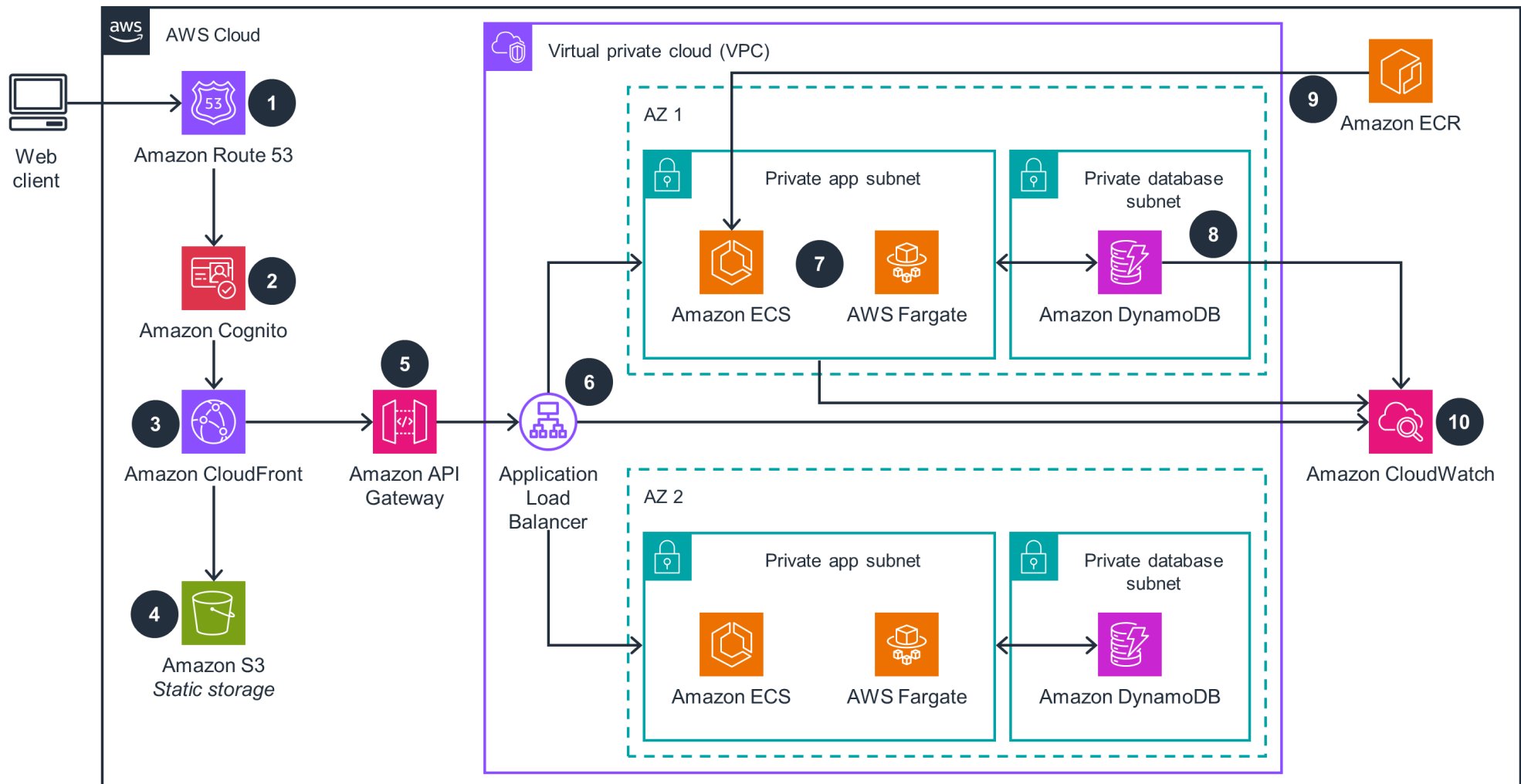


Consumer and social



Example of cloud web application

- A scalable, secure 3-tier web application using containerization on AWS



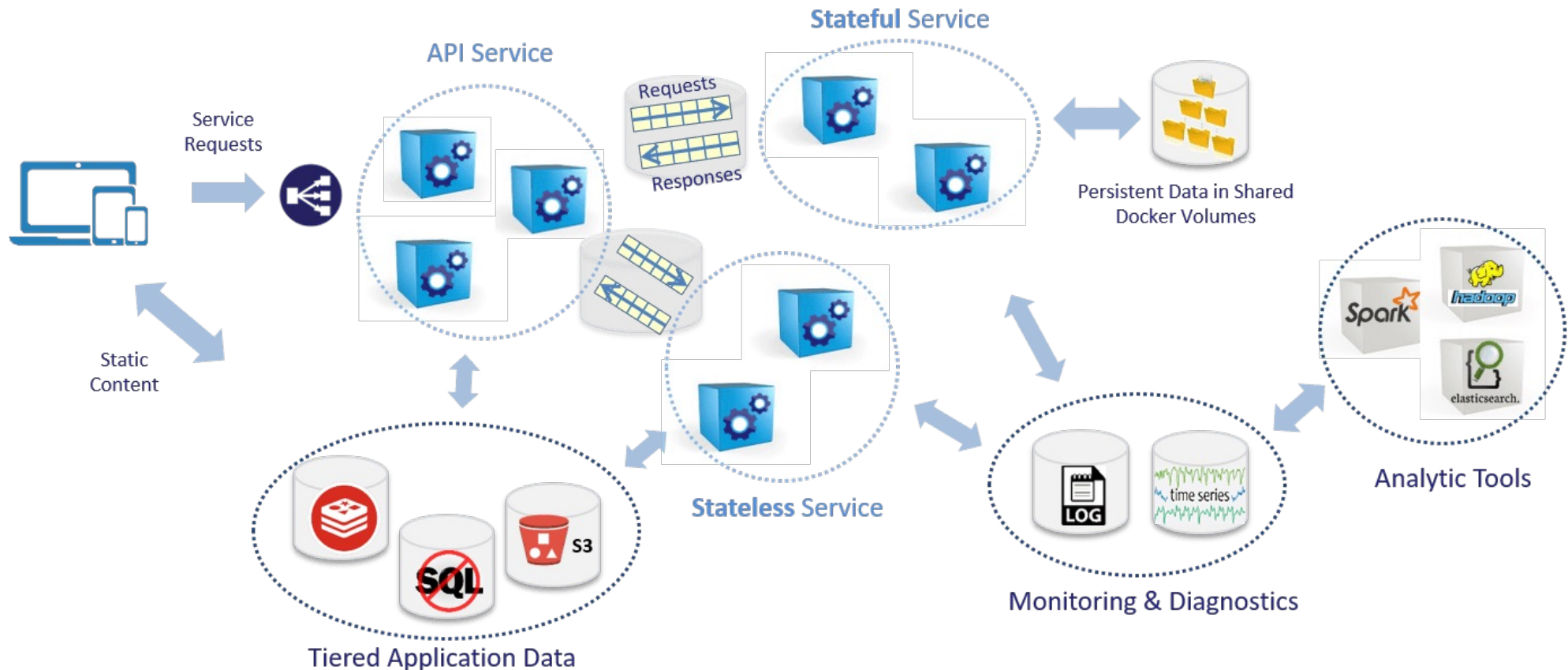
<https://aws.amazon.com/it/solutions/guidance/building-a-containerized-and-scalable-web-application-on-aws>

Cloud-native applications

- Cloud native: sw approach of building, deploying, and managing modern applications in cloud environments
- Cloud-native applications are a collection of **small**, **independent**, and **loosely coupled** services
- Goals: agile, highly scalable, flexible, and resilient applications
- We will study **microservices** and **serverless** applications

Cloud-native applications: high-level example

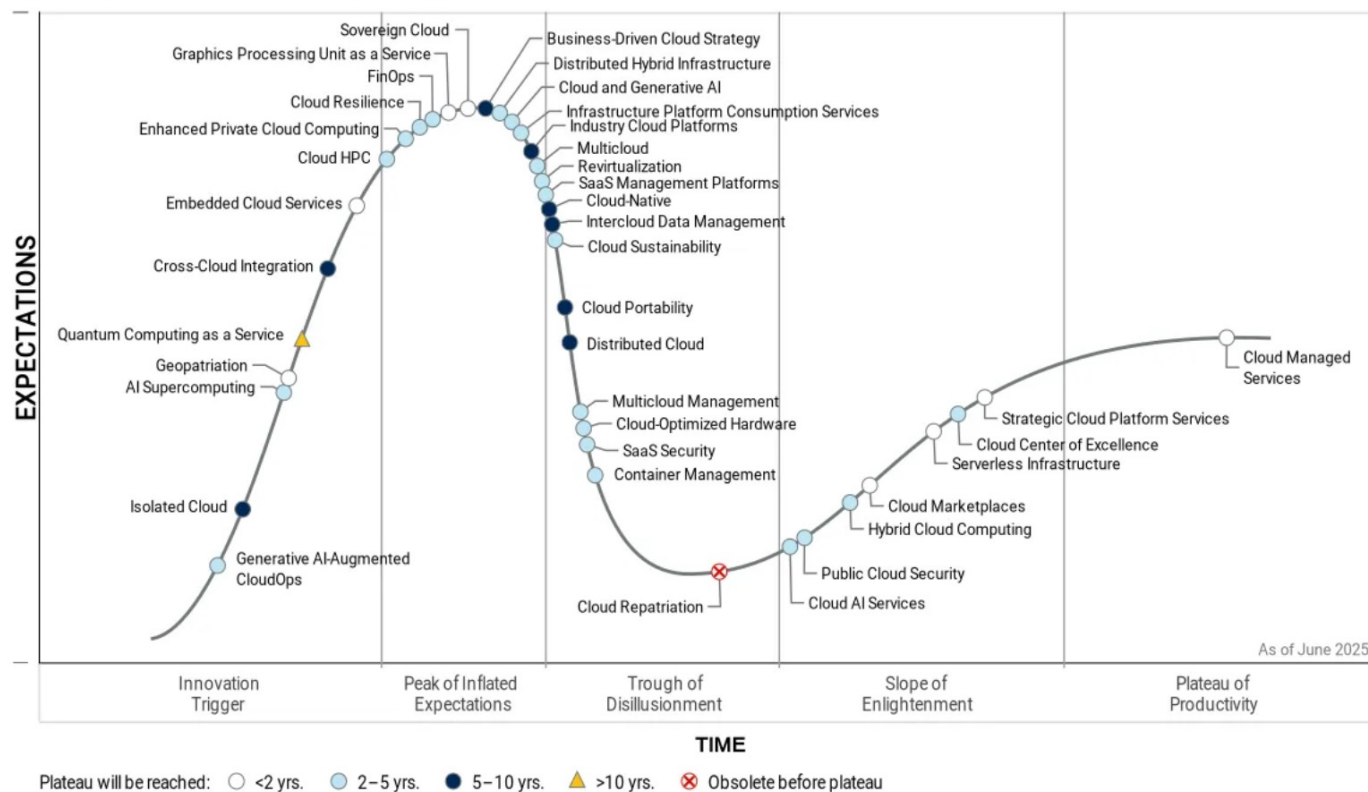
Stateful service: retain data or context between sessions or interactions



Cloud trends

- Hybrid and multi-cloud deployment models
- Generative AI and ML services, Container-as-a-Service and serverless
- Cloud governance and FinOps

Hype Cycle for Cloud Computing, 2025



Multi-cloud: challenges

- Multi-cloud is complex
 - Managing resources across different cloud providers
 - Automating workload placement, configuration and maintenance, including identity management and data protection/encryption
 - Monitoring performance, resource usage, and costs across multiple infrastructures and applications
 - Tracking changes in cloud providers' services and pricing (FinOps)
 - Predicting performance, resource usage and costs
- How to deploy multi-cloud applications?
 - Commercial purpose-built products
 - **Infrastructure as Code (IaC)**: tools like Terraform and Ansible allow you to define cloud infrastructure as code and deploy it across multiple cloud providers

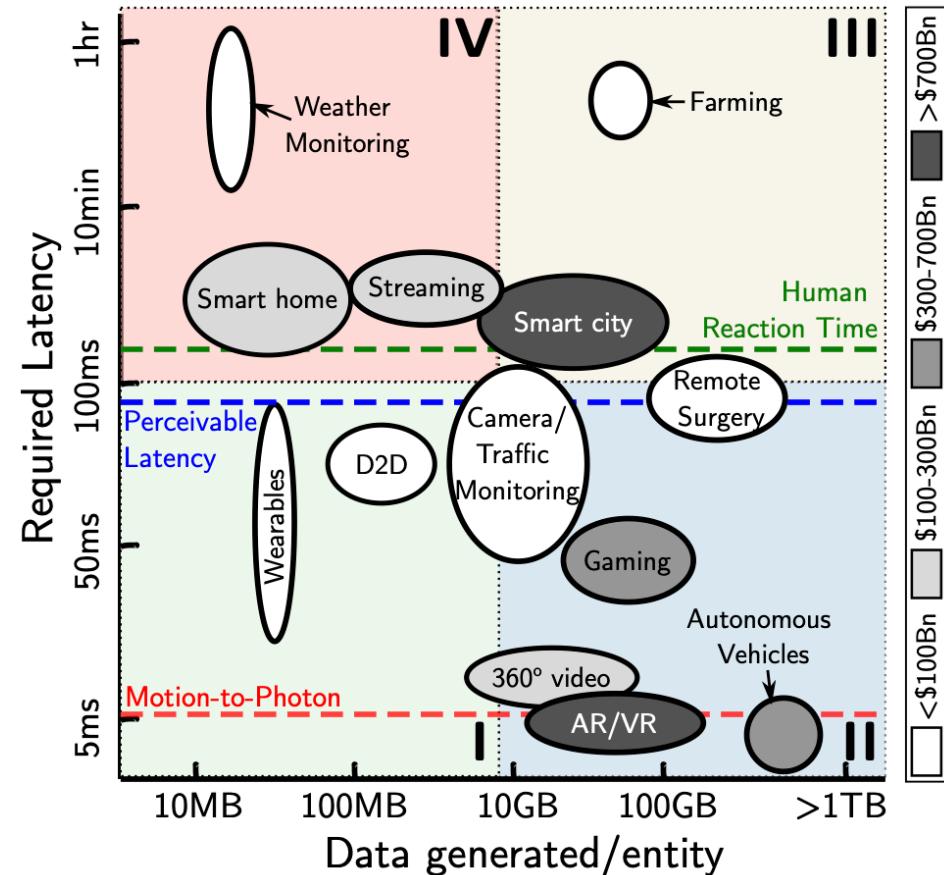

Hands-on

Cloud trends: applications

- Convergence of different technologies, evolving landscape
- Emerging applications
 - **Generative AI and AI-native Cloud**: inference, agentic workflows
 - **Metaverse**: collective virtual spaces, industrial digital twins
 - **Web3 and decentralized architectures**: focus on scaling and privacy, including layer-2 solutions and zero-knowledge proofs
 - **Quantum computing**: cloud-accessible Quantum-as-a-Service environments for cryptography, optimization, and molecular simulation

Does cloud suit every application?

- Different classes of applications with different requirements
- Challenge with **latency-sensitive and location-aware** applications
 - E.g., real-time manufacturing, self-driving vehicles, drones, cognitive assistants generate massive data volumes



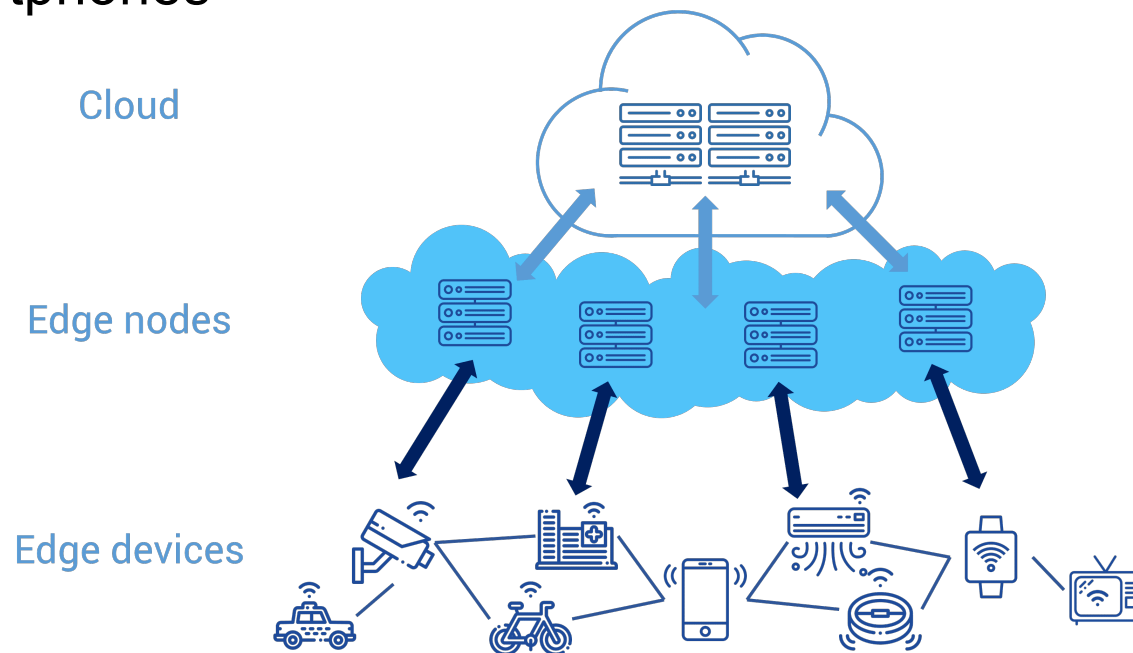
Recall: **latency is not zero!**

Beyond cloud: new computing paradigms

- Idea: **move** computation and storage to **nearby resources** located at the **edges** of Internet
- Evolution of new computing paradigms
 - **Edge computing**
 - **Computing continuum**
- Not apart from Cloud, but rather integrated with Cloud

Edge computing

- Shift computation and storage from “centralized” hyperscale cloud data centers to **proximity** of **edge devices** (e.g., IoT sensors), as close as one hop
 - **Edge nodes**: wide range of hardware, from lightweight IoT/edge gateways and micro-servers to mobile devices and smartphones

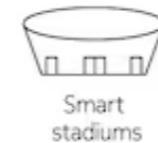


Edge computing

- Main benefits:
 - ✓ Reduce network latency and bandwidth requirements
 - ✓ Save energy
 - ✓ Improve privacy (e.g., on-premises compute and storage)
 - ✓ Bring AI and analytics where data are produced and consumed
 - ✓ Help in remote areas with poor network connectivity
- Strong relationship to 5G
 - 5G: ultra-low latency (1 ms), high-speed data transmission (10 Gbps), massive device connectivity, robust connectivity
- Strong relationship to AI: edge AI
 - Running ML, DL, and small language models directly on edge nodes

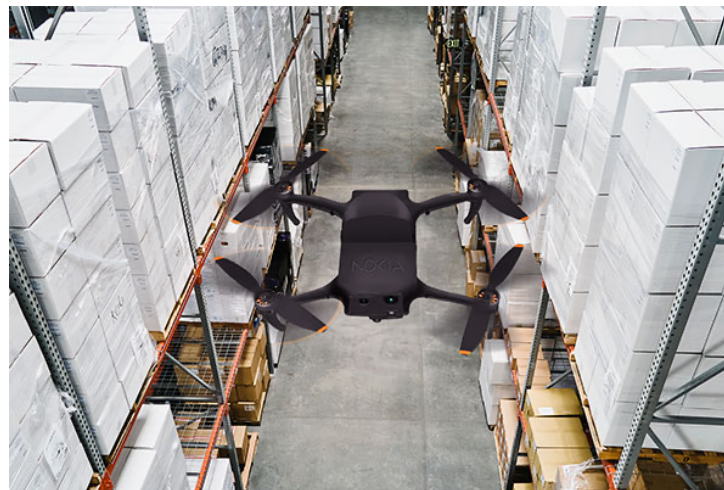
Edge computing: use cases

- Use cases
 - Almost all industry verticals and consumer habits
 - IoT applications
 - Industry 4.0 (e.g., predictive maintenance)
 - Smart cities
 - Extended reality (e.g., interactive shopping)
 - Autonomous vehicles
 - Healthcare
 - Interactive games



Example: Noika AIMS

<https://bit.ly/4yY1EGn>



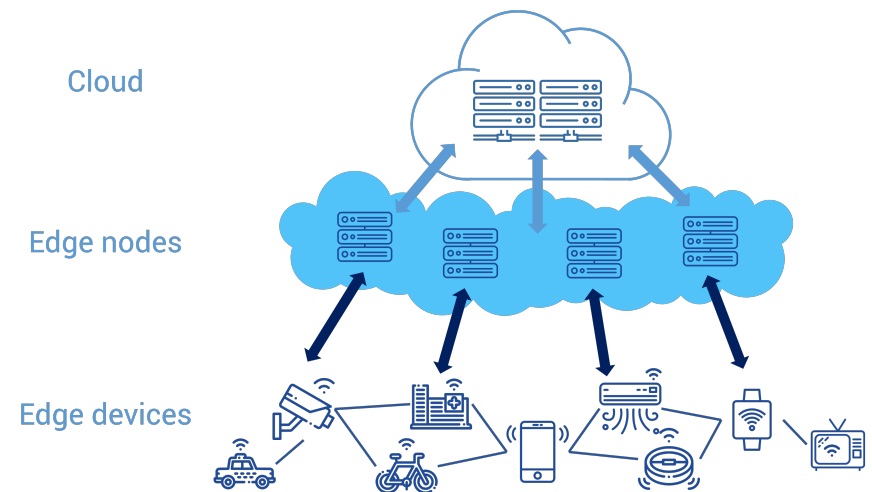
Edge computing: providers

- Cloud providers offering resources at the edge
 - AWS Local Zones, Azure Extended Zones, Google Distributed Cloud
- CDN and serverless edge compute providers
 - Cloudflare Workers, Akamai EdgeWorkers
- Edge hardware
 - Dell EMC Edge Gateway, NVIDIA Jetson and IGX, HPE Proliant Compute edge
 - Includes ruggedized products for harsh environments

Edge computing

- Some limitations
 - Resource constraints: edge nodes often have limited processing power and storage
 - Scalability challenges, e.g., scaling across many edge nodes
 - Security risks: more vulnerable and less secure environments
 - Limited standardization: protocols and frameworks
- How to address?
- (Partially) **offload** computation and storage to powerful **Cloud** servers

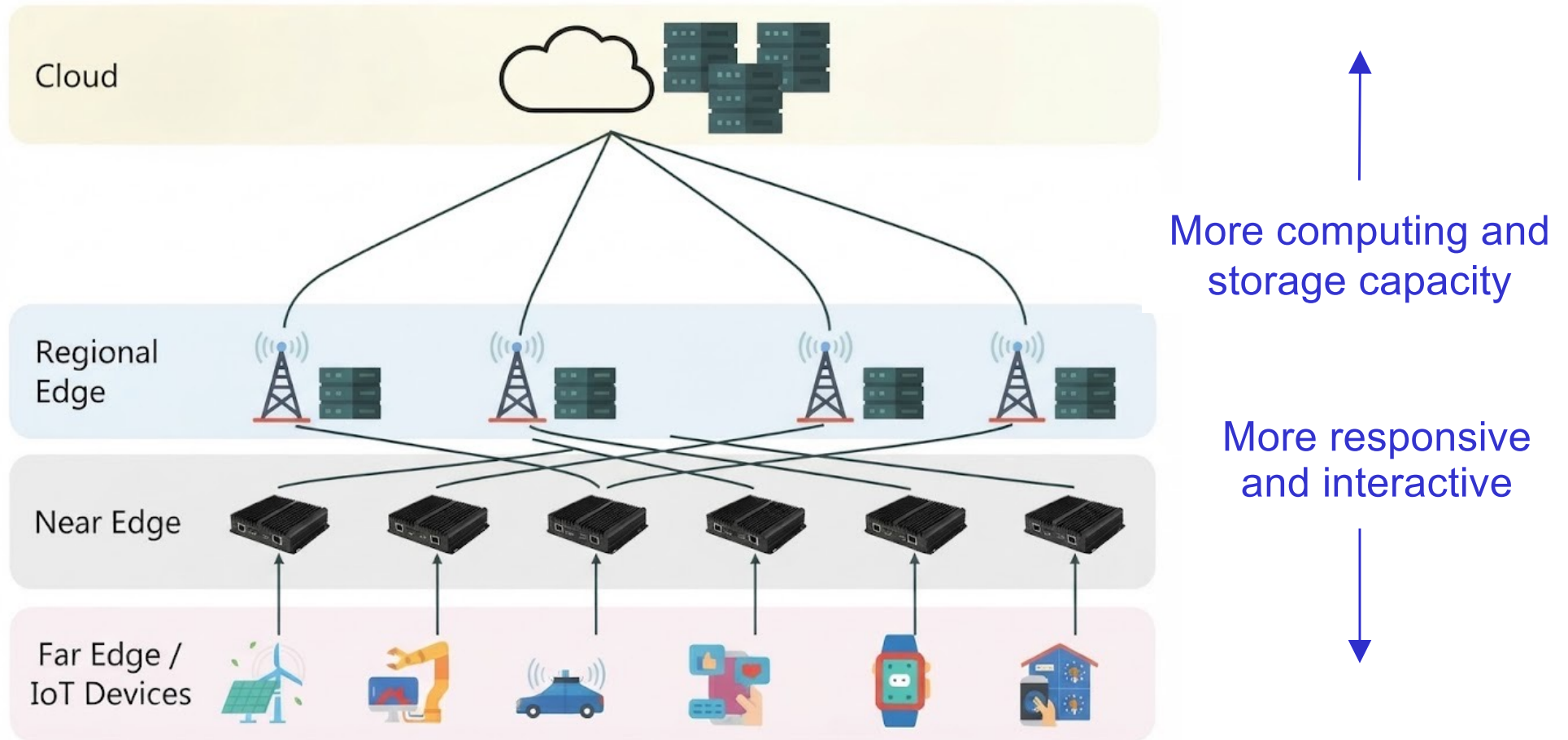
Is it enough?



Multi-layer edge computing

- Multiple edge layers (bottom to top)
- **Far edge / IoT devices**
 - Sensors, cameras, smartwatches, actuators, vehicles, and smart home devices
- **Near edge**
 - Smaller, localized hardware: micro servers, on-premise gateways
 - Located at: building, factory, cell tower base
- **Regional edge**
 - Larger facilities: micro data centers, telco data centers
 - Located at: telecom central offices, 5G base station aggregation sites, regional metropolitan hubs
 - Aka **fog computing**

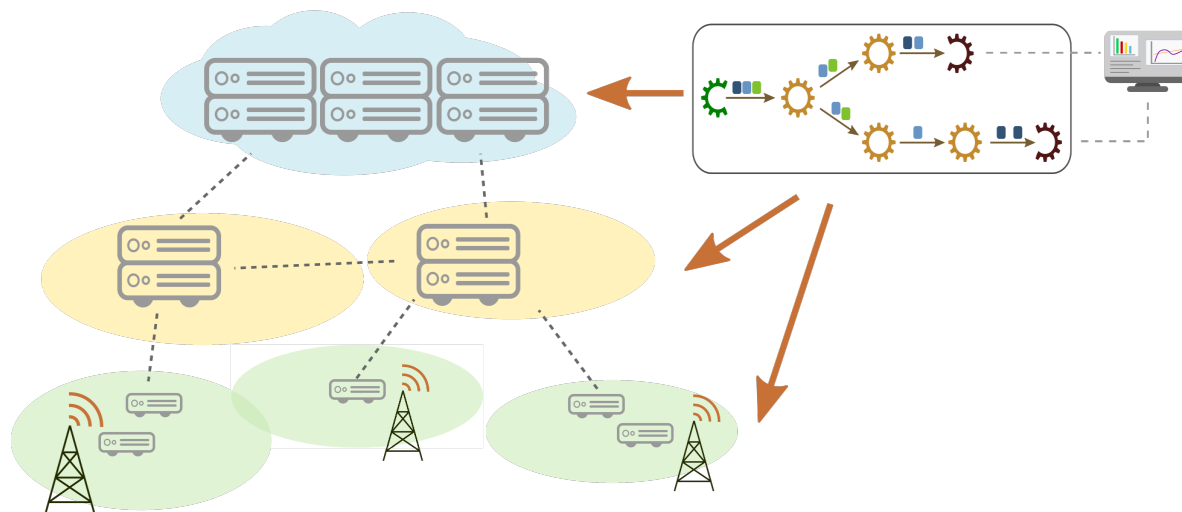
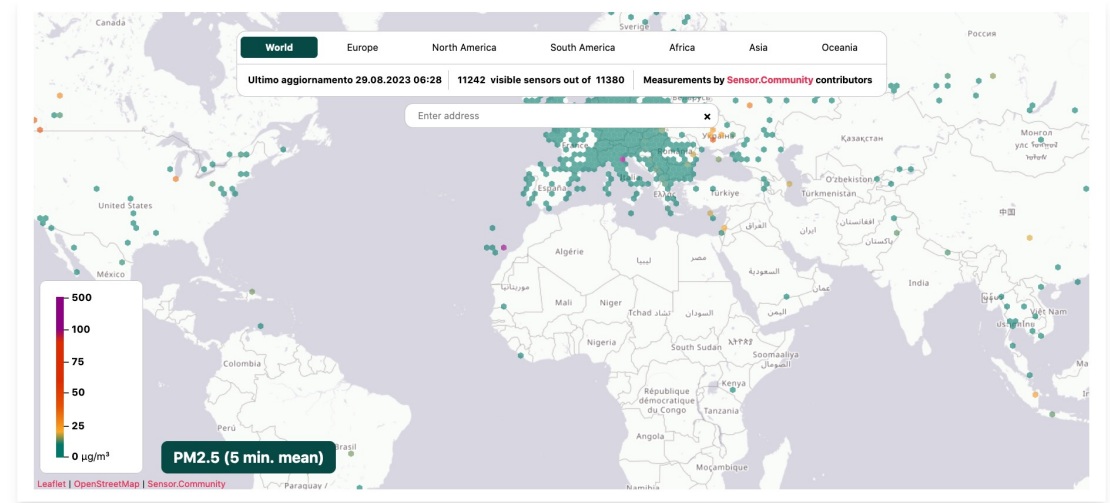
Putting all together: computing continuum



Example of computing continuum application

- Real-time analytics for environmental monitoring

Globally distributed environmental sensors



DAG-based applications (data streaming, microservices, serverless, ML pipeline, ...)

Computing continuum: challenges

- Infrastructure interoperability and seamless integration
- Unified programming abstractions
- Resource heterogeneity (hardware, energy, network bandwidth and latency)
- Performance and SLA guarantees
- Security and privacy
- Methodologies, programming models, tools, and platforms for computing continuum are **still in their infancy**

Summing up: cloud benefits

- Scalability, elasticity, flexibility
- Anytime and anywhere accessibility
- Improved security
 - E.g., 3.47 Tbps DDoS attack from 10,000 servers successfully stopped in 2022 <https://tinyurl.com/bdec45dy>
- Organizational and operation agility
- Simplified management
 - E.g., software updates and versioning
- No advanced payment
 - Costs scale with use
 - CapEx to OpEx
- Rapid business innovation
- Increase in productivity

IT benefits

Business benefits

Issues for cloud customers

- **Privacy, security and legal issues**
 - Control over data: location of data centers, data access
 - **Data sovereignty**: data are subject to the laws and governance structures of the nation where they are collected
 - Data encryption, also in transit
 - Data integrity, tracing and recovery
 - GDPR compliance (accountability, privacy by design, privacy by default)
- We had “stormy” clouds
 - “You have zero privacy anyway. Get over it.” (Scott McNealy, co-founder of SUN, 1999)
 - “If you have something that you don’t want anyone to know, maybe you shouldn’t be doing it in the first place.... The reality is that search engines do retain information... It could become available later...” (Eric Schmidt, Google CEO, 2009)

Issues for cloud customers

- Now?
 - EU codes of conduct: used by Cloud providers to demonstrate and verify GDPR compliance
 - CISPE Code of Conduct: for IaaS providers
<https://www.codeofconduct.cloud>
 - EU Cloud Code of Conduct <https://eucoc.cloud>
 - Be aware of privacy policies

"We collect location information when you use our services, which helps us offer features like driving directions, search results for things near you, and ads based on your location. Depending on the products you're using and settings you choose, Google may use different types of location information to help make some services and products you use more helpful." (Google, 2026) <https://policies.google.com/privacy?hl=en-US>

"Microsoft uses the data we collect to provide you with rich, interactive experiences. We use data to: provide our products, ..., improve and develop our products, ..., advertise and market to you... We also use the data to operate our business, which includes analysing our performance, meeting our legal obligations, developing our workforce, and doing research." (Microsoft, 2026) <https://privacy.microsoft.com/en-gb/privacystatement>

Issues for cloud customers

- **Network latency**

- ms latency from users and devices to cloud data centers
- How to address? Computing continuum

- **Cloud portability**

- Customer's ability to move and adapt their applications and data between their own systems and cloud services, or across different cloud providers and deployment models
- Risks of vendor lock-in
 - E.g., specific features offered by PaaS provider make it difficult to switch to another provider
 - E.g., difficulty in extracting data and migrating it to different provider
- Towards portability
 - Containers: enable applications to run consistently across different cloud environments
 - Cloud automation tools (IaC tools)

Issues for cloud customers

- **Cloud interoperability**
 - Ability of public cloud services, private cloud services, and cloud service customer's system to understand and work with each other (compatibility of interfaces, configuration, authentication and authorization, methods data formats, ...)
 - **Standards** for cloud portability and interoperability
 - ISO/IEC 19941:2017, IEEE 2302-2021 Standard for Intercloud Interoperability and Federation
 - European initiative GAIA-X <https://gaia-x.eu>
- **SLA negotiation and management**
 - Lack of SLIs based on end-user experience
 - SLA monitoring often on customers' shoulders

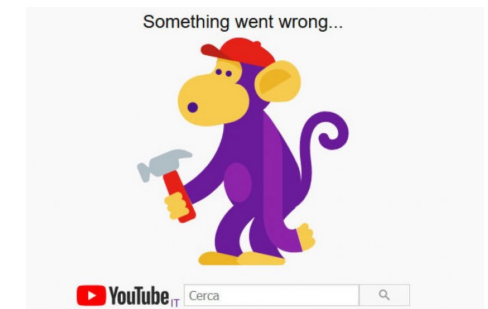
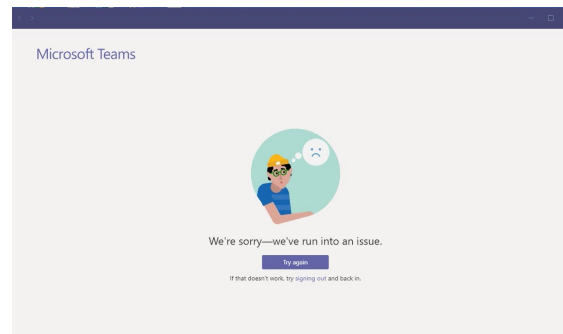
Issues for cloud customers

- The two faces of scalability and elasticity
 - Can your application scale?
 - Design applications to leverage cloud capabilities
 - We will see architectural patterns (microservices and containers, stateless architecture, ...), auto-scaling policies, automated orchestration
 - Does cloud provider support real elasticity, i.e., without performance degradation?
 - Scaling latency depends on virtualization technologies
 - Providers impose limits and quotas on resources (e.g., maximum number of VMs, storage size)
 - Stateful workloads are more complex to scale (require distributed replication, data consistency)

Issues for cloud customers

- **Cloud outages:** examples

- 2020: AWS Kinesis went down, impacting other AWS services using it (**cascading failure**); **poor sw design** (one connection per OS thread) led to resource exhaustion <https://aws.amazon.com/message/11201>
- 2023: Google Cloud services in Europe hit by outage when Paris' data center caught **fire** <https://thenewstack.io/google-cloud-services-hit-by-outage-in-paris>
- 2024: Microsoft cloud services, including Azure and Teams, experienced a global disruption due to a **defective update** from CrowdStrike, a cybersecurity company <https://tinyurl.com/4tt3ex6p>
- 2025: Google Cloud outage affected Gmail, Drive, Meet, and third-party platforms like Spotify and Discord. Cause: technical failure in Google Cloud's API management system, leading to **invalid quota updates** <https://tinyurl.com/4v7musa8>
- Examples in 2026 <https://bit.ly/4xVYrGw>



Issues for cloud providers

- **Increasing energy consumption**

<https://doi.org/10.1038/d41586-025-01113-z>

- **Uncertainty in service demands**

- Example of provider-side solution: spot instance market to sell unused Amazon EC2 capacity

- **SLA management**

- Cascading effect

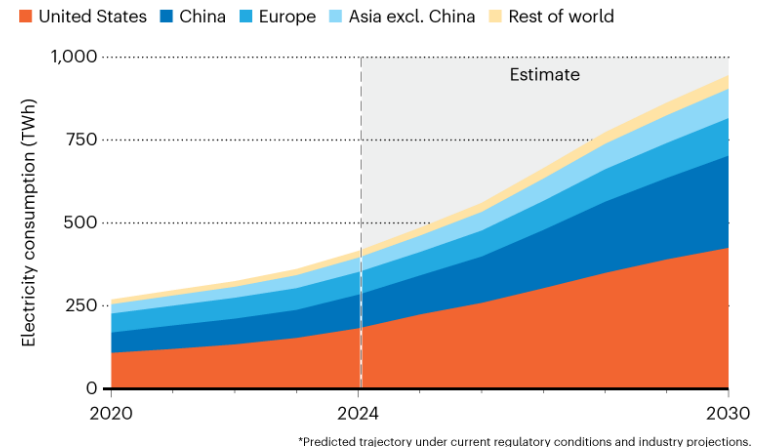
- **Cloud interoperability**

- Interoperable tools with standardized interfaces

- **Lack of skilled expertise**

DATA-CENTRE ENERGY GROWTH

China and the United States are predicted to account for nearly 80% of the global growth in electricity consumption by data centres up to 2030*.



©nature

Summing up

- Cloud computing is:
 - Here to stay and one of the most hyped topics in IT
 - Essential part of modern IT
 - Changing how digital services are designed, built, and delivered
 - Key driver of digitalization across society
- However, cloud and its evolutions are complex systems
 - Heterogeneous resources, many technologies, plethora of applications with heterogeneous workloads and complex interactions
 - All difficult aspects of large-scale distributed systems
 - Recall: “old” problems become new problems on a totally different scale and open to new solutions

References

- Chapters 1 and 2 of Marinescu book
- A view of cloud computing <https://doi.org/10.1145/1721654.1721672>
- The NIST definition of cloud computing
<https://doi.org/10.6028/NIST.SP.800-145>
- Cloud SLAs: Present and future
<https://doi.org/10.1145/2331576.2331586>
- Elasticity in cloud computing: What it is, and what it is not
https://www.usenix.org/system/files/conference/icac13/icac13_herbst.pdf
- The state of the art in locally distributed web-server systems
<https://doi.org/10.1145/508352.508355>
- A manifesto for future generation Cloud Computing: Research directions for the next decade <https://doi.org/10.1145/3241737>
- Edge computing: Vision and challenges
<https://doi.org/10.1109/JIOT.2016.2579198>
- The computing continuum: Past, present, and future
<https://doi.org/10.1016/j.cosrev.2025.100782>