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. 2024/25

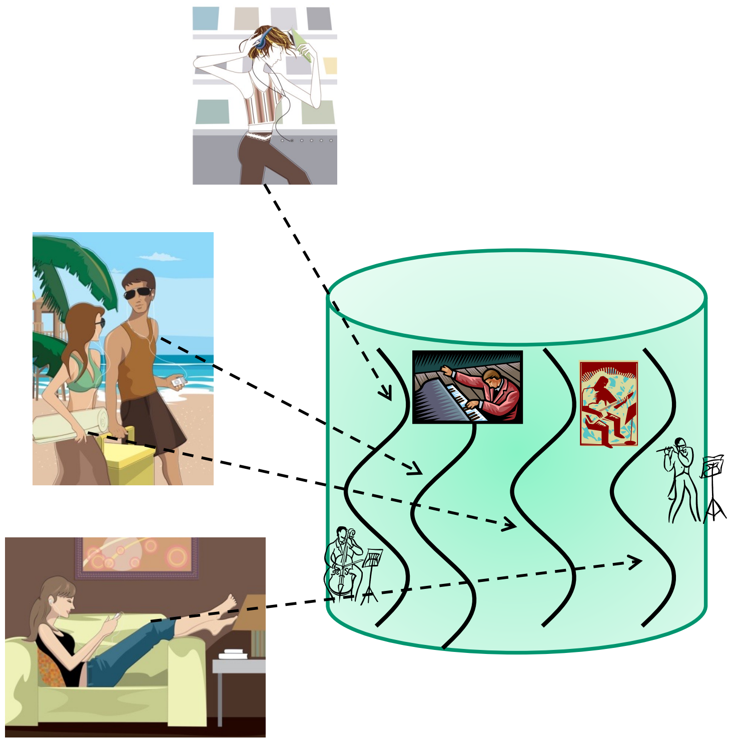
Valeria Cardellini

Laurea Magistrale in Ingegneria Informatica

A simple app: classic solution

- A simple application: video playback
- How to scale it?

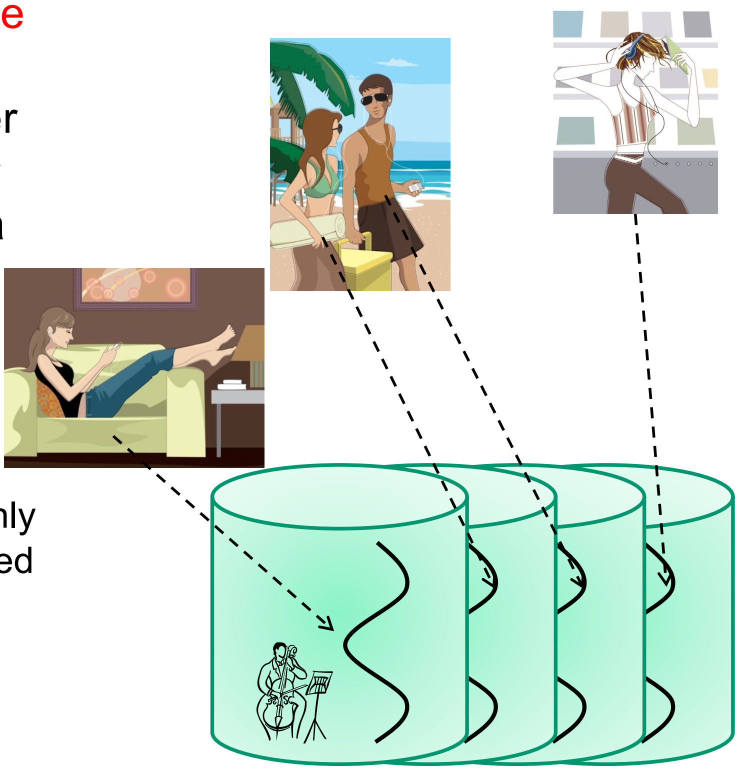
- "Classic" solution: multithreaded application that exploits multicore parallelism
- Cons:
  - ✗ Complex
  - ✗ Thread crash may impact many users
  - ✗ Scalability limits



A simple app: cloud-native solution

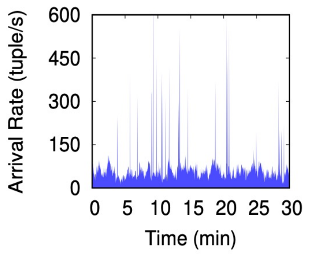
- A simpler cloud-native solution: a single-threaded video server instantiated once per user and running in a sw container

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



The real problem: scale and complexity

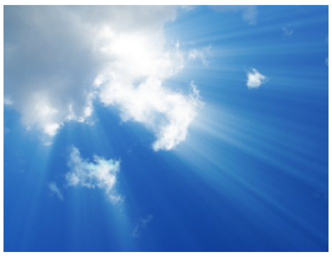
- How to realize sw services that:
  - Handle millions of requests per second
  - Manage workload fluctuations of one order of magnitude (or even more) in a quite short period
  - Store EBs of data
    - 1 EB = 2^{60} B = 10^{18} B



- 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…

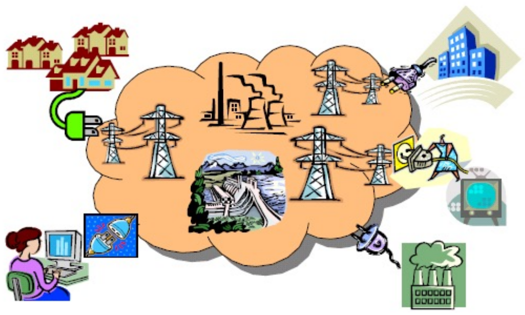
- Water

- 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.

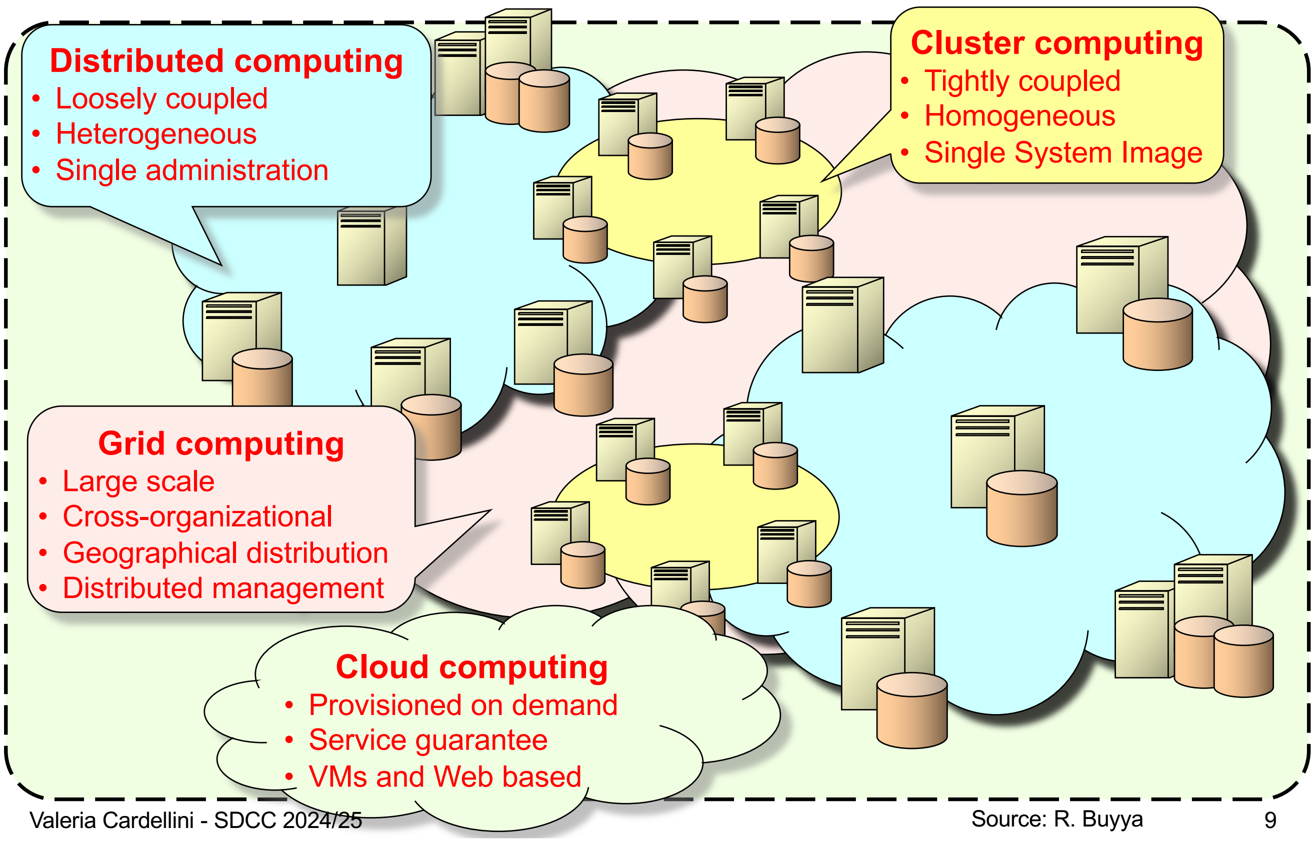
Utility computing: new idea?

- 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, Univ. Melbourne
  - “Cloud is the computer”

How do computing paradigms differ?



Source: R. Buyya

Cloud computing

Cloud → "Cloud" is the graphic symbol we use to represent Internet

computing → Computing includes:
- Computation
- Storage
- Coordination

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

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.

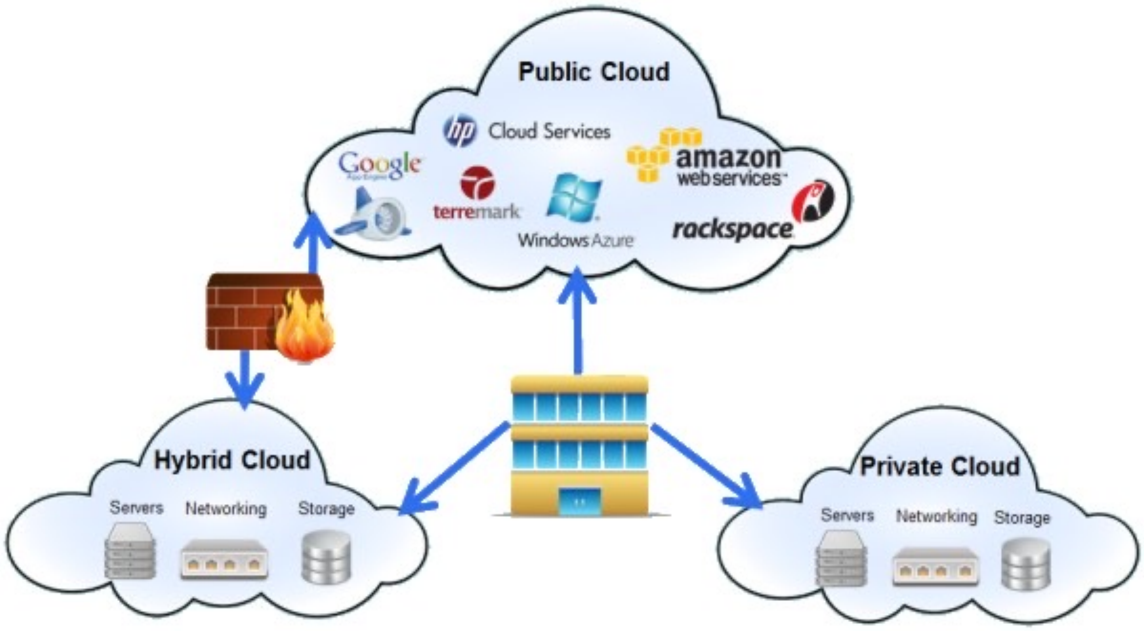
… that share some essential characteristics

- **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

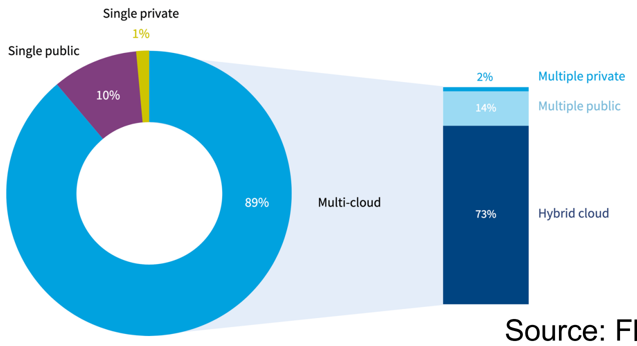
- Public cloud
  - Cloud infrastructure: provisioned for open use by the general public (*multi-tenant*)
  - Owned, managed, and operated by a business, academic, or government organization, or some combination of them
  - Exists on premises of cloud provider
  - Services can be free or fee-based

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)
  - Owned, managed, and operated by the organization, a third party, or some combination of them
  - 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



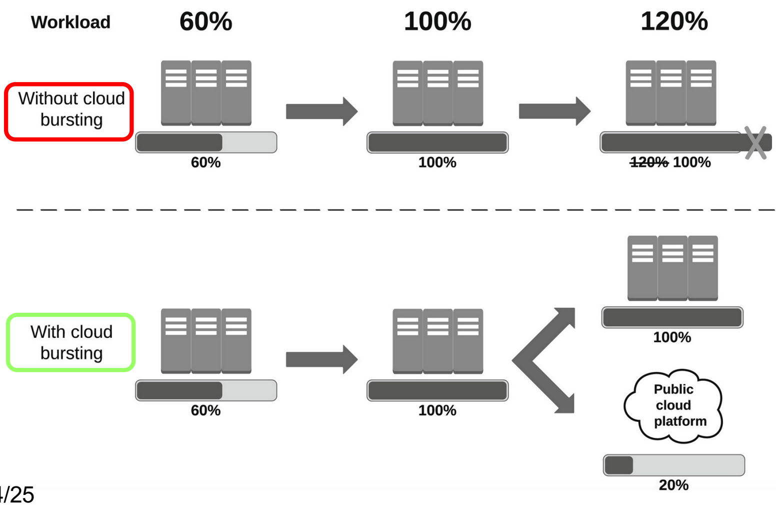
Source: Flexera 2024 Cloud Report

Deployment models: hybrid

- Pros
  - ✓ Balance resources and costs
  - ✓ Increase flexibility
    - Differentiate privacy, e.g., sensitive data is kept in private cloud
    - Differentiate services: apps are siloed on different clouds
    - Improve availability: public cloud for disaster recovery in case of unexpected outage
    - Improve geographic distribution and compliance: public cloud resources in different geographic regions
    - Improve performance: by means of cloud bursting
  - ✓ Reduce vendor lock-in risks
    - Vendor lock-in happens when someone is forced to continue using a service regardless of quality, because switching away from that service is not practical

Hybrid cloud: cloud bursting

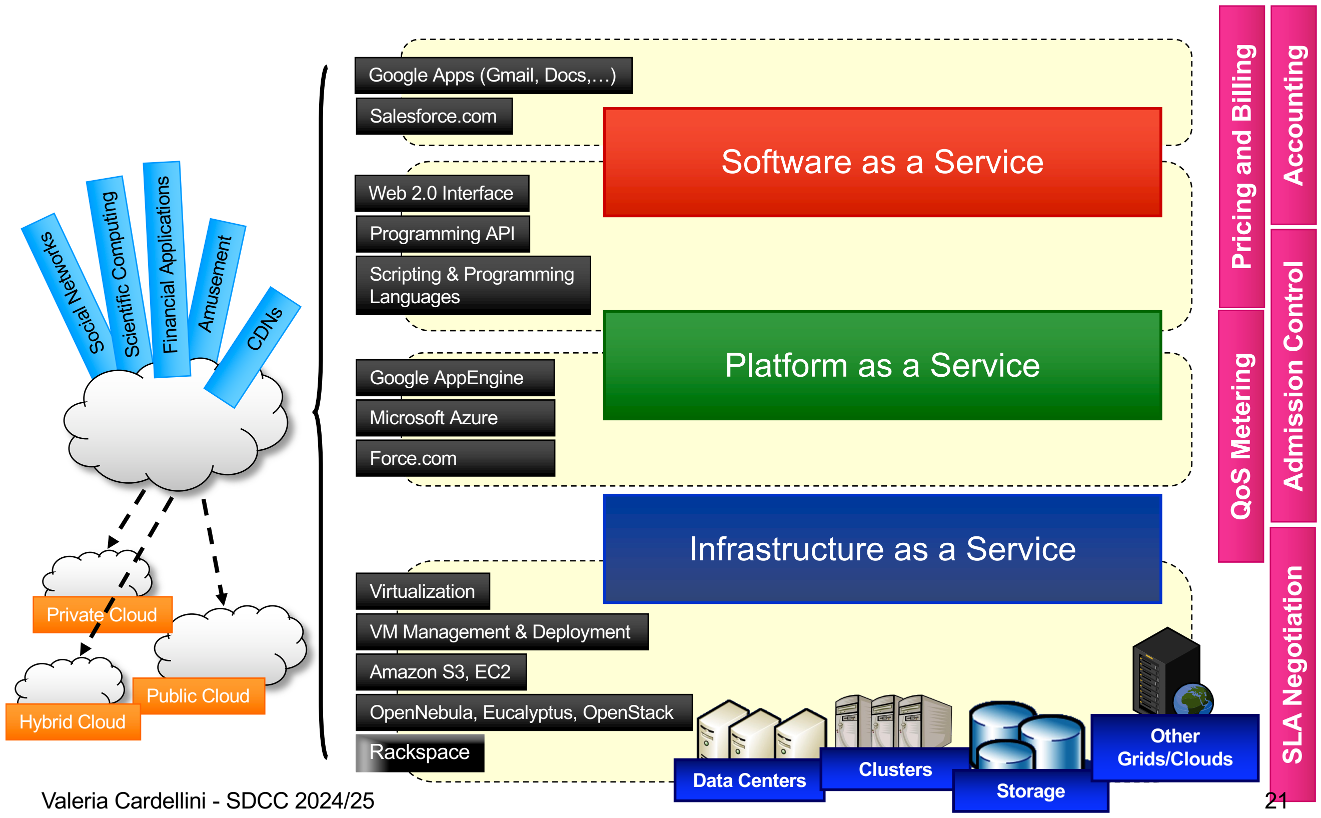
- Use hybrid cloud dynamically to manage workload spikes when private cloud capacity is insufficient
  - Workload runs in a private cloud which reaches its peak capacity
  - Use public cloud to handle workload spikes without maintaining excess capacity on-premise



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, Facebook, Netflix, Twitch, and Zoom use commercial clouds
- Cloud market size
  - Reached $670 billion in 2023
  - Expected to reach $1400 billion by 2028

Cloud stack: basic service models



Service models: IaaS

- Infrastructure as a Service (IaaS)
  - Compute, storage and network resources as services
  - Customer capability
    - To provision fundamental computing 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)

IaaS: examples

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

IaaS: AWS EC2

- Let us launch a virtual machine on EC2 from AWS Management Console
- Few steps in 2-3 minutes:
  1. Sign in to AWS Management Console https://signin.aws.amazon.com/
  2. Open Amazon EC2 console by choosing EC2 under Compute
  3. Choose AWS Region (e.g., EU Frankfurt)
  4. From EC2 dashboard, choose Launch Instance
  5. Choose software configuration (operating system, application server, and applications) from available Amazon Machine Images (AMIs)
     - AMI provided by AWS, user community, or AWS Marketplace
  6. Choose instance type and configure it
     - T-shirt sizes: various combinations of CPU, memory, storage, and networking capacity
  7. If not yet configured, select security group to open a specific network port (e.g., SSH) for the launched instance
  8. Select an existing key pair or create a new one to securely connect to the instance after it launches

Service models: PaaS

- Platform as a Service (PaaS)
  - Platform that allows customers to develop, run and manage scalable applications, without the complexity of building and maintaining the underlying infrastructure
  - Customer capability
    - To develop, deploy and test onto the cloud (consumer-created or consumer-acquired) 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, Microsoft Azure App Service
- Managed Kubernetes platforms
  - Google Kubernetes Engine, Amazon Elastic Kubernetes Service, Azure Kubernetes Service
- Platforms for entire application lifecycle
  - Heroku, Red Hat OpenShift, VMware Tanzu
- Salesforce Platform
- Serverless computing/Function as a Service (FaaS)
  - AWS Lambda, Google Cloud Functions, IBM Functions

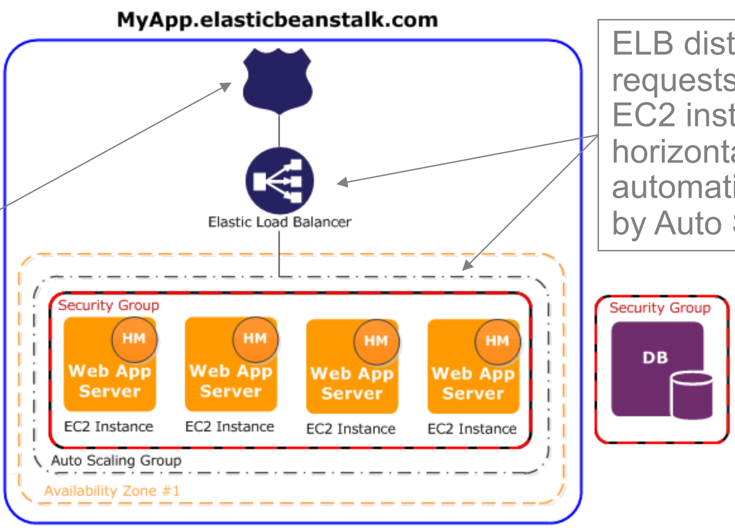
PaaS: AWS Elastic Beanstalk

- 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

Also Amazon Route 53 to map the app URL to the URL of the front-end load balancer; can also be used for geographic distribution among different app replicas



ELB distributes requests among the EC2 instances, whose horizontal scaling is automatically managed by Auto Scaling

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

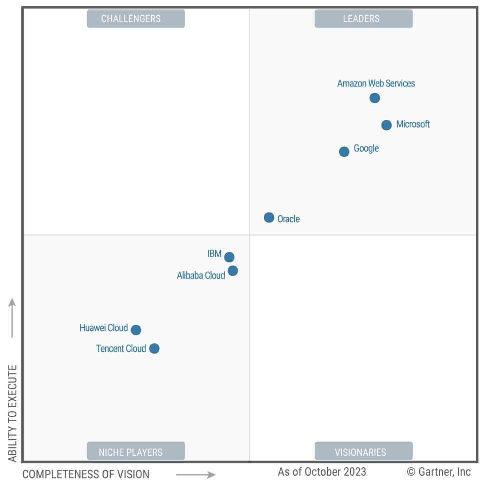
IaaS vs. PaaS

- 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 (2023)

Figure 1: Magic Quadrant for Strategic Cloud Platform Services



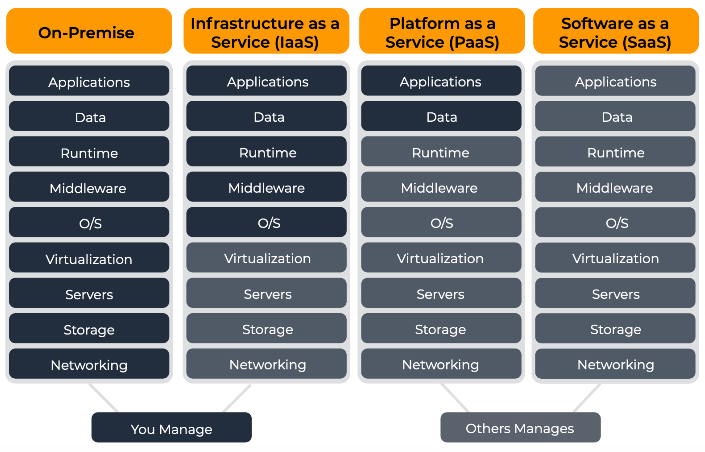
Service models: SaaS

- 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



- Additional service models, e.g.:
  - Database as a Service (DBaaS)
  - Function as a Service (FaaS)
  - Monitoring as a Service (MaaS)
  - … Anything as a Service (XaaS)

Cloud pricing models

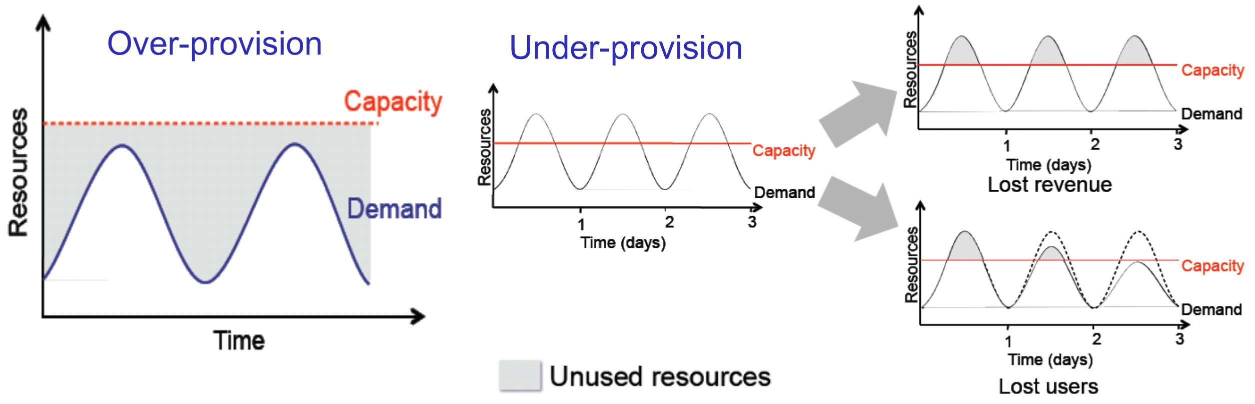
- Cloud providers offer a number of pricing models
- Pay-per-use / on-demand
  - Users are charged based on cloud resources usage
  - Pricing granularity depends on service, e.g., by hour for AWS EC2, by ms for AWS Lambda
  - ✓ Maximum flexibility: no need to plan in advance
- Fixed / reserved
  - Users are charged a fixed amount per month for cloud resources
  - ✓ Significant discount compared to on-demand pricing
  - ✗ Upfront payment

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

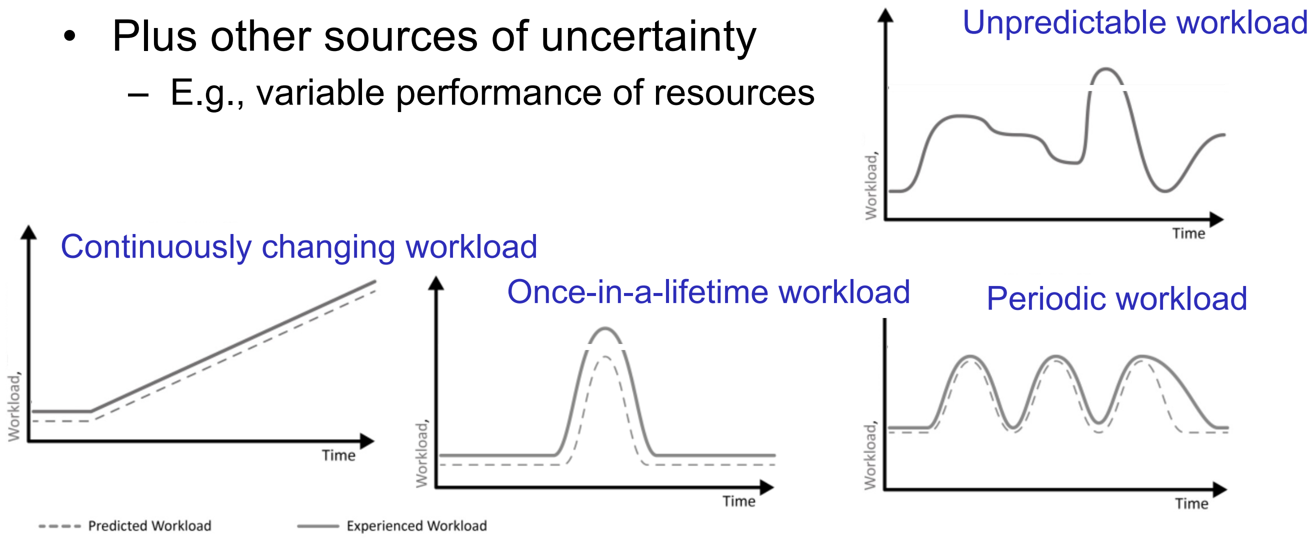
Capacity planning and resource provisioning

- 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)



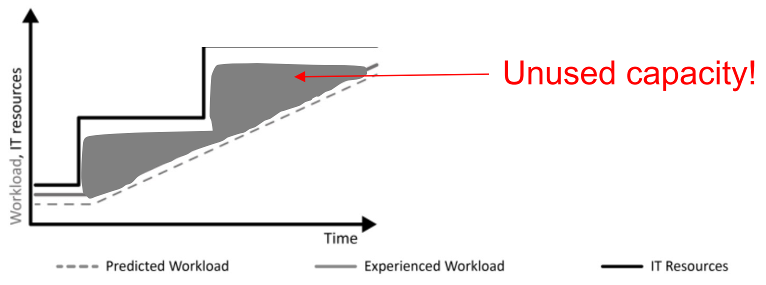
To make things more complicated: uncertainty

- Fluctuating workloads with different patterns
- Plus other sources of uncertainty
  - E.g., variable performance of resources



A solution…

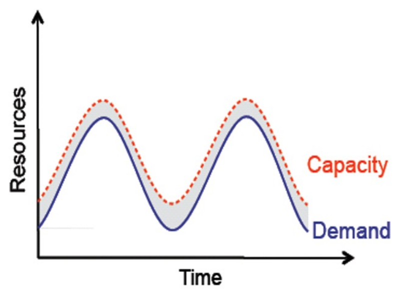
- Forecast demand
- Manual dynamic provisioning: scale-out(in) and/or scale-up(down) *manually* IT resources

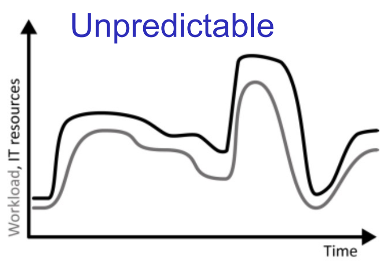


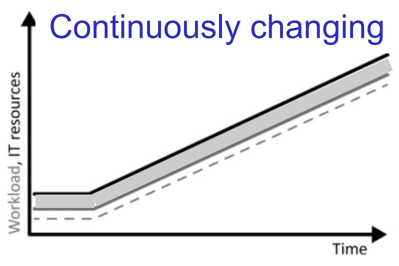
- Not practical!

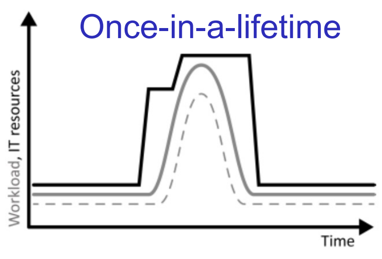
The solution: elastic scaling (aka auto-scaling)

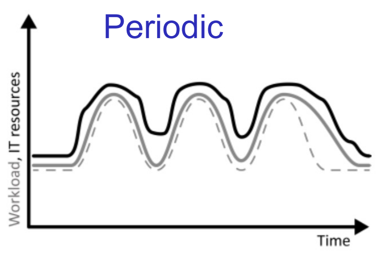
- Let's automatically align IT resources to changing workload





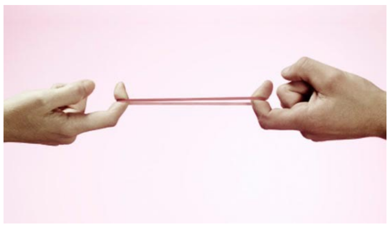






Elasticity

- (Physics) Property of returning to an initial form or state following deformation
  - ➢ Stretch when a force stresses it
  - ➢ Shrink when the stress is removed

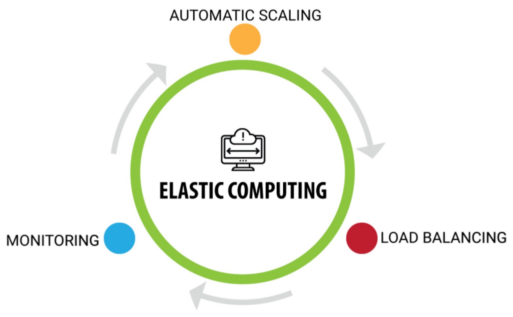


- (Computer science) Degree to which a computing system is able to adapt to workload changes by provisioning and de-provisioning resources in an autonomic manner, such that at each point in time the available resources match the current demand as closely as possible

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

Elasticity and Cloud computing

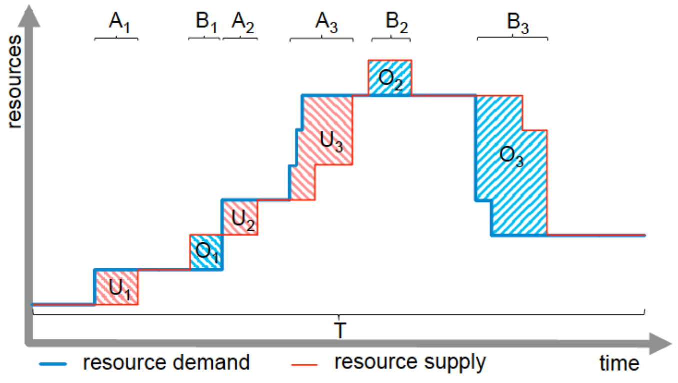
- Elasticity: key feature of **Cloud computing**
- How do we achieve elastic computing?
- We need a self-adaptive software system



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

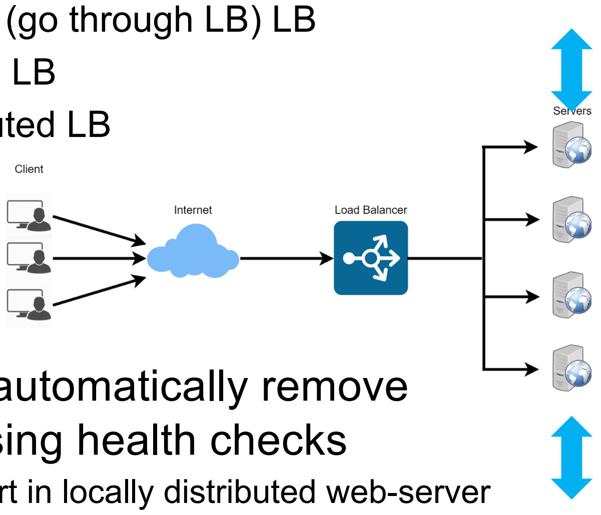
- Consider over-provision and under-provision



- 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 external IP address (VIP) assigned to LB
  - Network (layer 4) or application (layer 7) LB
  - One-way (responses from replicas go directly to clients) or two-way (go through LB) LB
  - Hardware or software LB
  - Centralized or distributed LB



- LB also detects and automatically remove unhealthy replicas using health checks

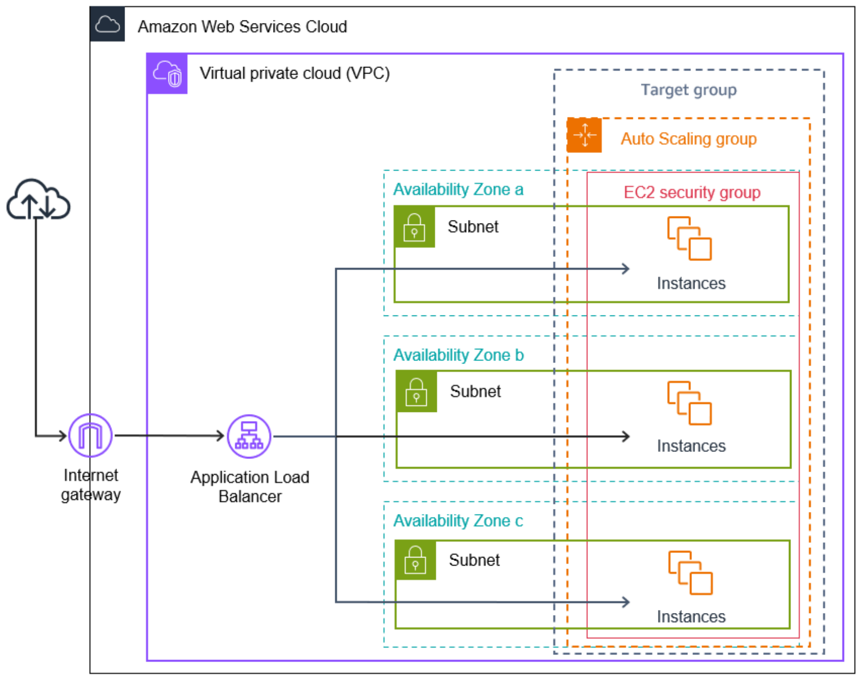
Cardellini et al., The state of the art in locally distributed web-server systems, ACM Comput. Surv., 2002

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 product: Envoy https://www.envoyproxy.io, nginx https://nginx.org/en/docs/http/load_balancing.html

Elasticity: architectural point of view

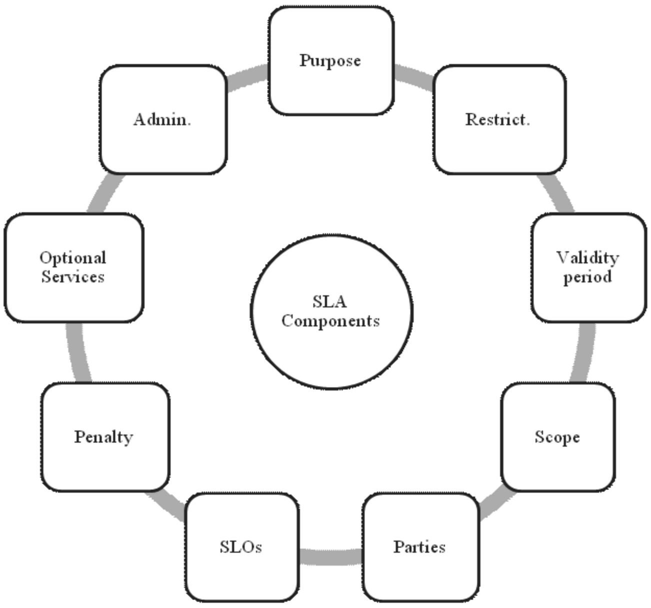
- Example of load balancing and auto-scaling using AWS



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

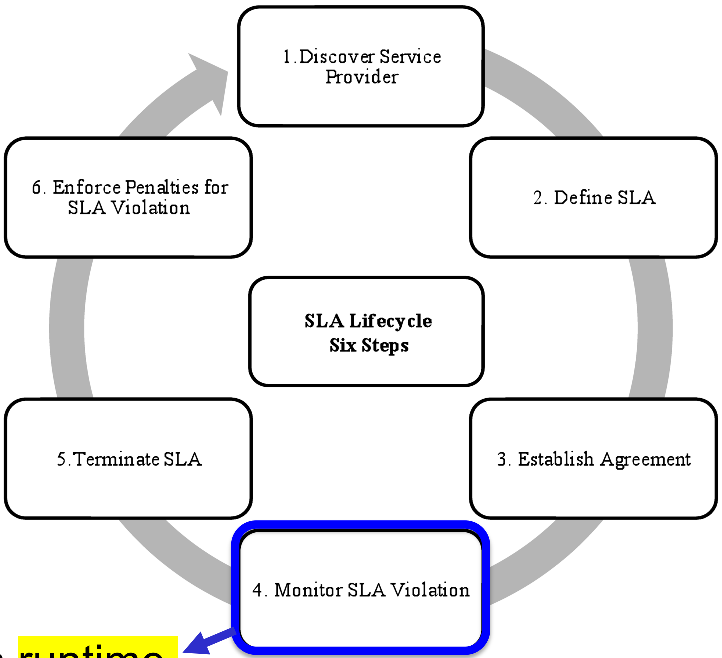
Service Level Agreement (SLA)

- 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): something you can measure
  - E.g., availability, response time
  - Availability: the most common SLI for cloud services

- Service Level Objective (SLO): a predicate over a set of SLIs
  - E.g., a fixed threshold on a single SLI

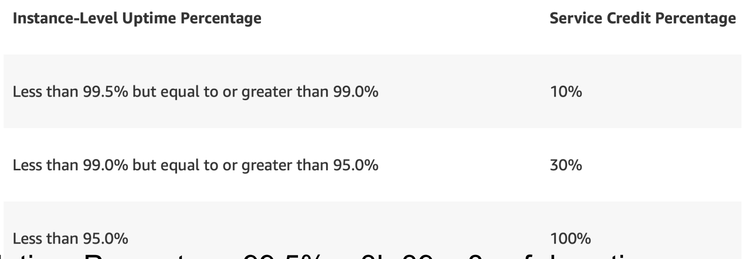
  monthly uptime percentage for a VM is at least 99.99%

↑ SLI ↑ SLO

Example: Amazon Compute SLA

- **Service Commitment** https://aws.amazon.com/compute/sla

  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

- **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

- Another example: https://cloud.google.com/compute/sla
- Issues related to cloud SLAs:
  - SLA jargon
  - Availability
    - Average over a period hides distinction between many short outages and a few long ones
  - No SLI expressed in terms of response time
  - Service credits only for future payments
  - Burden of detecting SLA violation on cloud customers
  - Time period for reporting SLA violation
  - Non-negotiable or customizable SLA for most users
  - How to compare SLAs of different cloud providers?

Cloud monitoring

- Cloud monitoring goal: track health of systems and services deployed in cloud
- Cloud monitoring tools
  - Allow cloud customers to collect and analyze data on system-oriented and application-oriented metrics from cloud resources and services

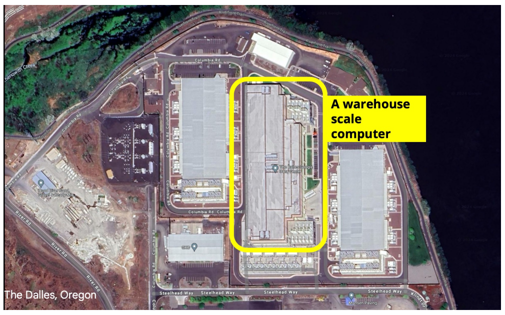
System-oriented monitoring metrics

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)

  - Built-in tools offered by cloud providers: e.g., Amazon CloudWatch https://aws.amazon.com/cloudwatch, Google Cloud Monitoring https://cloud.google.com/monitoring
  - Third-party tools: e.g., Datadog https://www.datadoghq.com, Dynatrace https://www.dynatrace.com, Prometheus https://prometheus.io

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

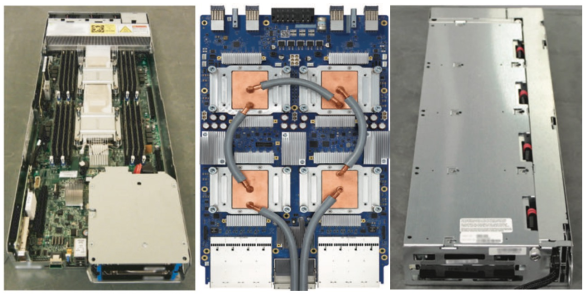


Source: John Wilkes (Google)

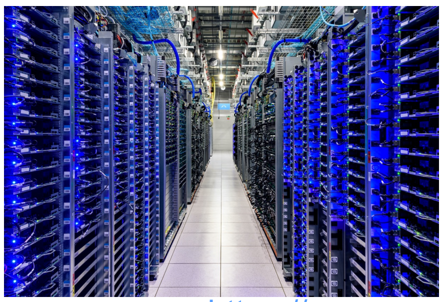
Barroso et al., The Datacenter as a Computer: Designing Warehouse-Scale Machines, 3rd ed., 2019 https://pages.cs.wisc.edu/~shivaram/cs744-readings/dc-computer-v3.pdf

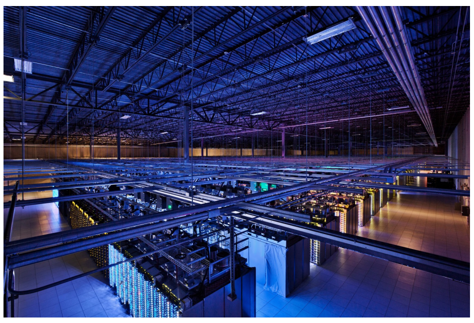
Inside a cloud data center

- Hardware building blocks



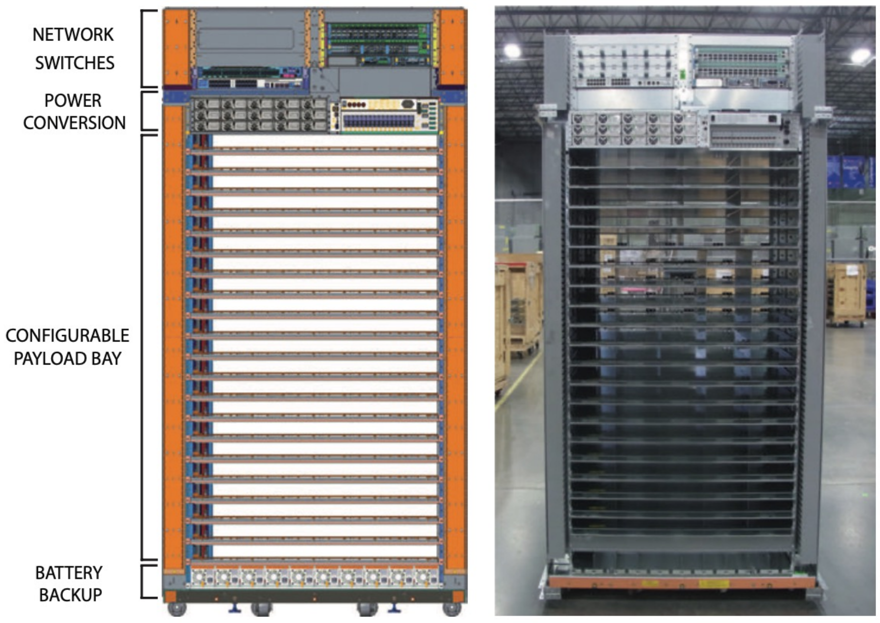
- … which are assembled into interconnected racks and rows





https://www.google.com/about/datacenters

Inside a cloud data center: a server rack



Outside a cloud data center

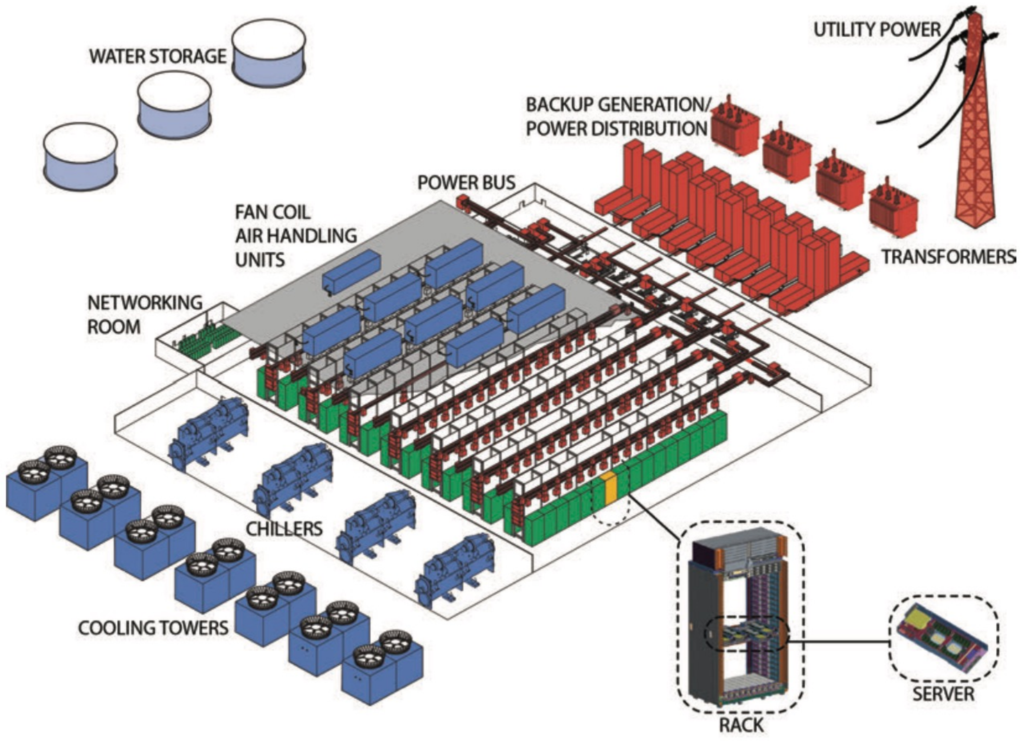
Cooling towers

- How large? ~100000 m^2
- How many servers? ~10^6
- Power-efficient?

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

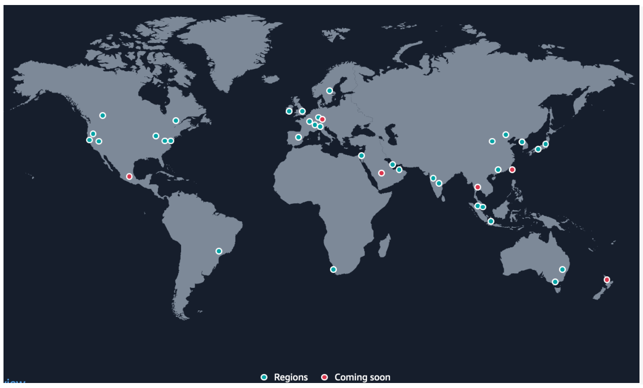
- E.g., average PUE for Google data centers is 1.1

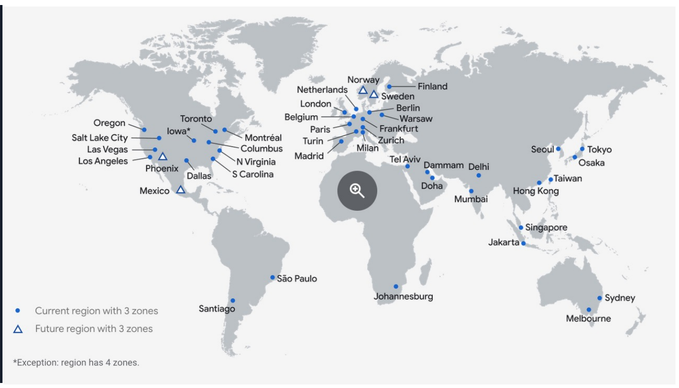
Main components of a cloud data center



Geographic distribution of cloud data centers

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





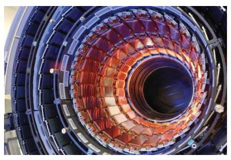
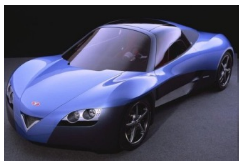
https://aws.amazon.com/it/about-aws/global-infrastructure https://cloud.google.com/about/locations

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

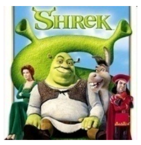
Cloud applications

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

Healthcare, scientific and technical

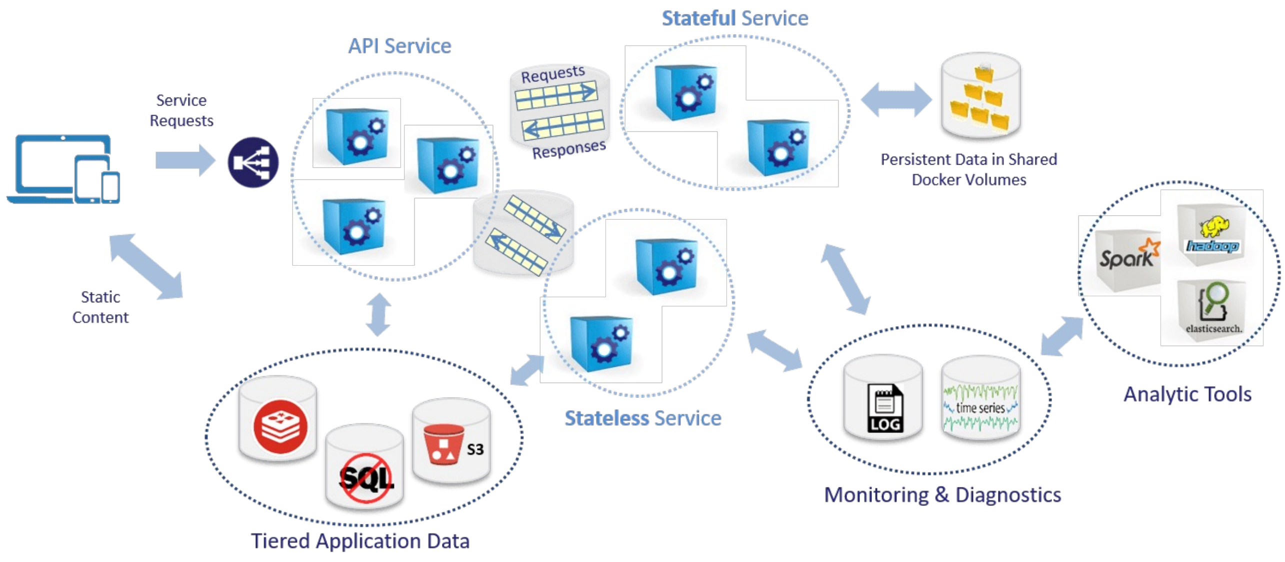
Consumer and social



Cloud-native applications

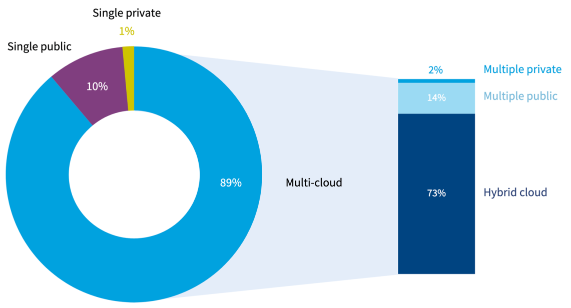
- Cloud native: sw approach of building, deploying, and managing modern applications in cloud environments
- Cloud-native applications are a collection of small, independent, and loosely coupled services
- Goals: agile, highly scalable, flexible, and resilient applications
- We will study microservices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

Cloud-native applications: high-level example



Cloud trends

- Public cloud adoption continues to accelerate
- Use of AI and ML, Container-as-a-Service, serverless is rising
  - 15% of on-premises production workloads will run in containers by 2026, up from less than 5% in 2022 [Gartner]
- Most organizations choose a multi-cloud strategy: concurrent usage of multiple clouds
  - Hybrid, multiple private, multiple public



Multi-cloud: challenges

- Multi-cloud management is complex:
  - Manage resources spread across different clouds
  - Automate workload placement, configuration and maintenance (including identity management and data protection/encryption)
  - Monitor performance of multiple infrastructures and apps
  - Assess changes in service portfolio and pricing models of cloud providers
  - Summarize resource usage and costs
  - Predict usage and costs
- Organizations can use
  - Commercial purpose-built products
  - Infrastructure-as-code (IaC) open-source tools to **automate configuration and deployment**: Chef, Puppet, Ansible and Terraform

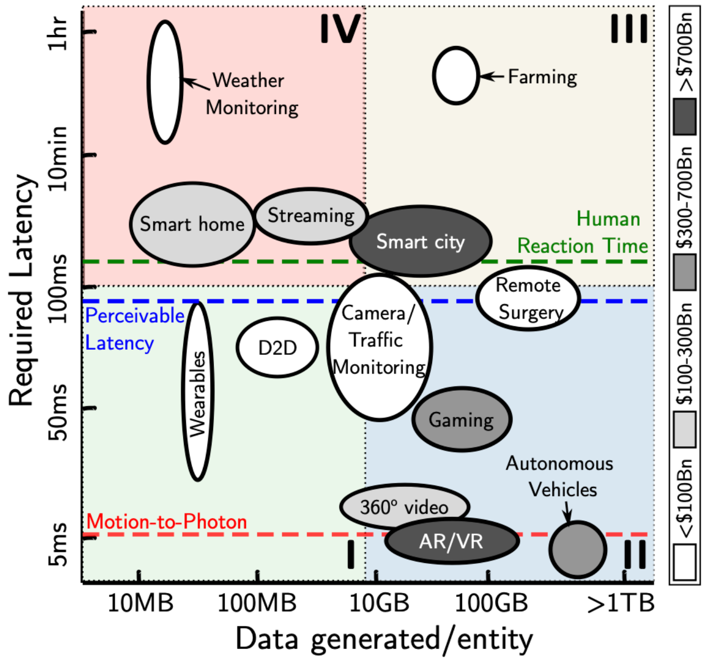
Hands-on

Cloud trends: applications

- Involve convergence of different technologies, all in cloud
- Current applications
  - Internet of Things
  - Big Data
  - AI and ML
- Emerging applications
  - Generative AI
  - Metaverse: collective virtual 3D shared space, created by the convergence of virtually enhanced physical and digital reality
  - Web3: new stack of technologies built on blockchain protocols that support the development of decentralized web applications and enable users to control their own identity, content, and data

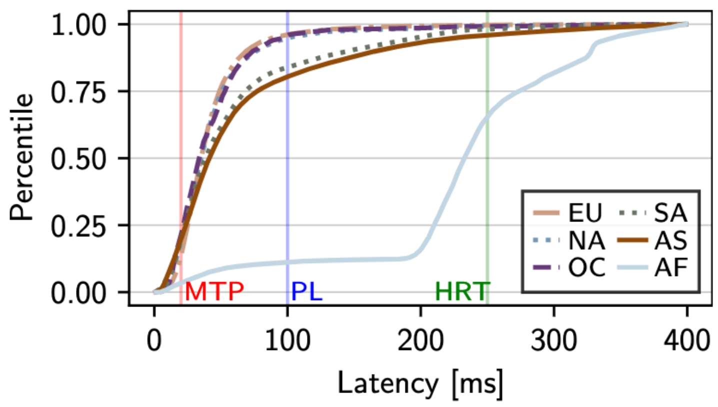
Does cloud suit for any application?

- Different classes of applications, different performance requirements



Away from cloud

- Some latency-sensitive and location-aware apps (e.g., real-time manufacturing, self-driving vehicles, flocks of drones, cognitive assistance) use large data volumes originated from devices and users
  - Cloud-only can be impractical due to network latency



Motion-to-Photon (MTP)
Human Perceivable Latency (PL)
Human Reaction Time (HRT)

RTT distribution to nearest cloud data center grouped by continent

Cloudy with a chance of short RTTs: analyzing cloud connectivity in the internet, IMC 2021 https://tinyurl.com/3wztmmz6

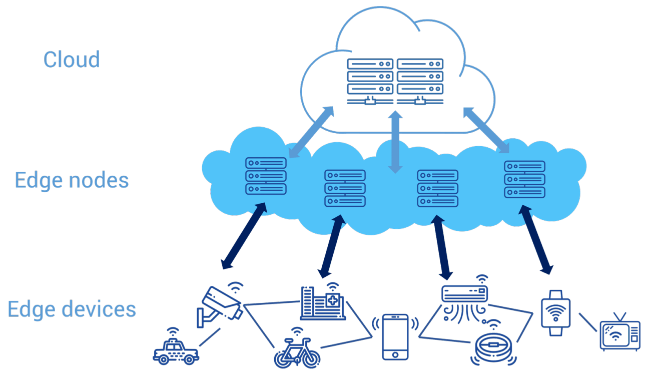
Away from cloud: new computing paradigms

- Idea: **move** computation and storage to nearby resources located at the edges of Internet

- Evolution of new computing paradigms
  - **Edge computing**
  - **Fog computing**
  - **Computing continuum**
- Not apart from Cloud, but rather integrated with Cloud

Edge computing

- Shift computation and storage from cloud to proximity of edge devices (e.g., IoT sensors), as close as one hop to them
  - Edge node: micro server, IoT/edge gateway, mobile device

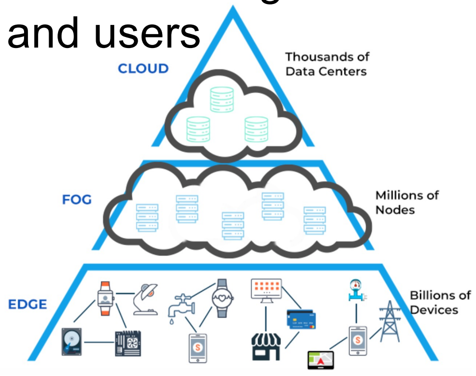


- Main benefits:
  - Reduce network latency and bandwidth requirements
  - Save energy
  - Improve privacy
  - Bring AI and analytics where data are produced and consumed
- Need to (partially) offload computation and storage to powerful Cloud servers

Is it enough?

Fog computing

- Layered model that pushes computation and storage from cloud down, close to data origin and users
  - Fog node: *micro* data center, located near edge layer and IoT devices
  - Fog nodes can be organized in a multi-level hierarchy



- Fog computing definitions
  - "A highly virtualized platform that provides compute, storage, and networking services between end devices and traditional cloud computing data centers, typically, but not exclusively located at the edge of network." (Bonomi et al., 2012)
  - "A horizontal, system-level architecture that distributes computing, storage, control and networking functions closer to the users along a **cloud-to-thing continuum**." (OpenFog consortium, 2017)

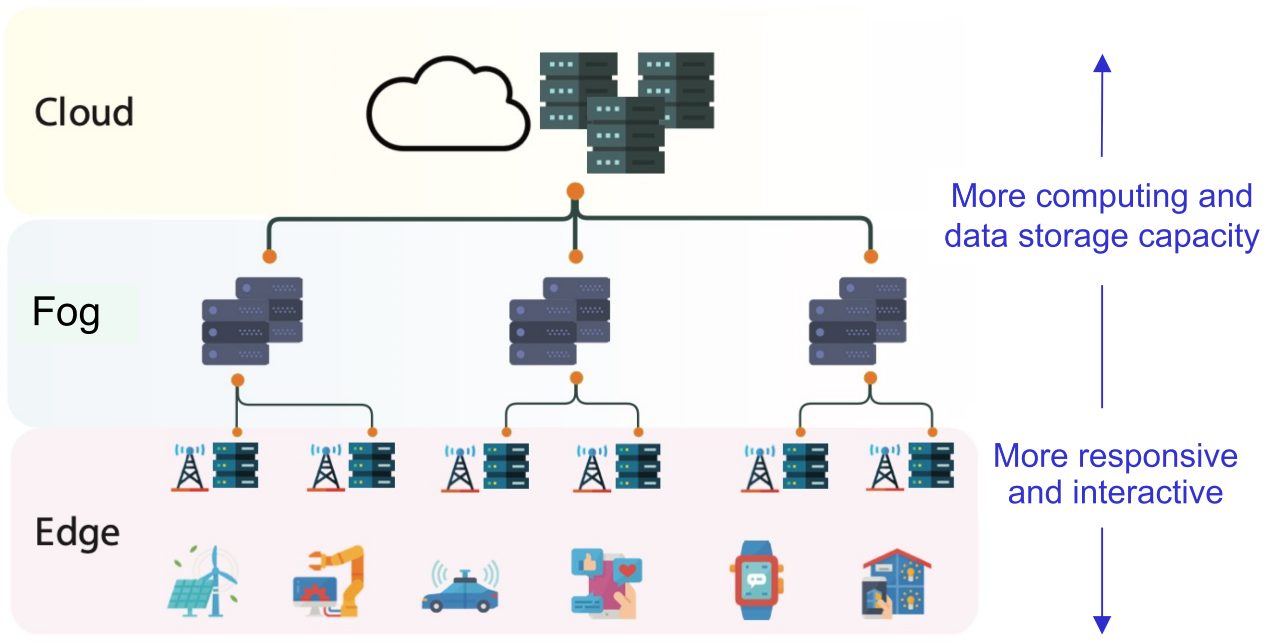
Fog vs. edge computing?

- Similar computing paradigms at network edges
  - Roots in content delivery networks and peer-to-peer networks
  - Decentralized architectures
  - Beneficial to low-latency apps, real-time analytics and AI
- Any difference?
  - No: interchangeable

    "Edge computing refers to the enabling technologies allowing computation to be performed at the edge of the network, on downstream data on behalf of cloud services and upstream data on behalf of IoT services." From “Edge computing: Vision and challenges” http://www.weisongshi.org/papers/shi16-edge-computing.pdf
  - Yes: Fog is more tightly integrated with cloud and its deployment size can be larger than edge

👉 Fog: layer in between edge and cloud

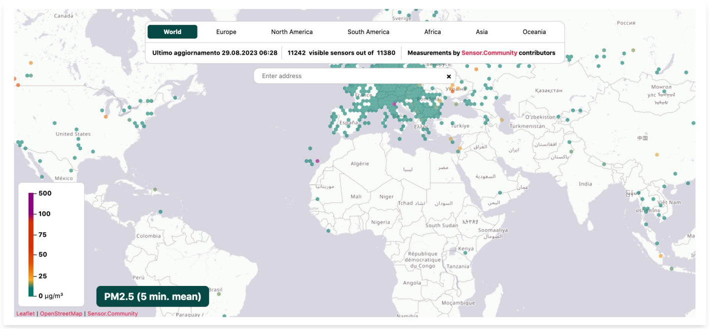
Putting all together: computing continuum


Cloud
Fog
Edge
More computing and data storage capacity
More responsive and interactive

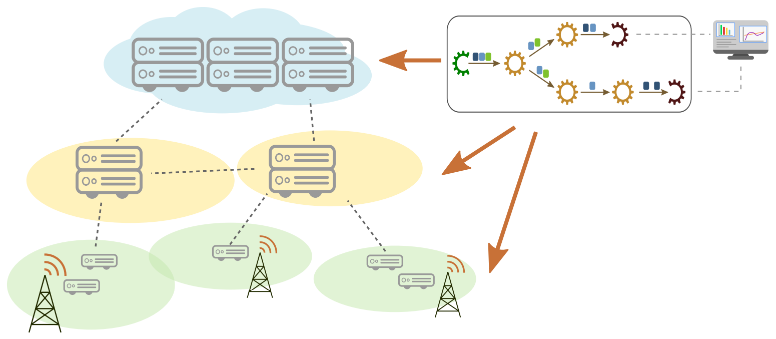
Putting all together: computing continuum application

- Real-time analytics for environmental monitoring

Globally distributed environmental sensors



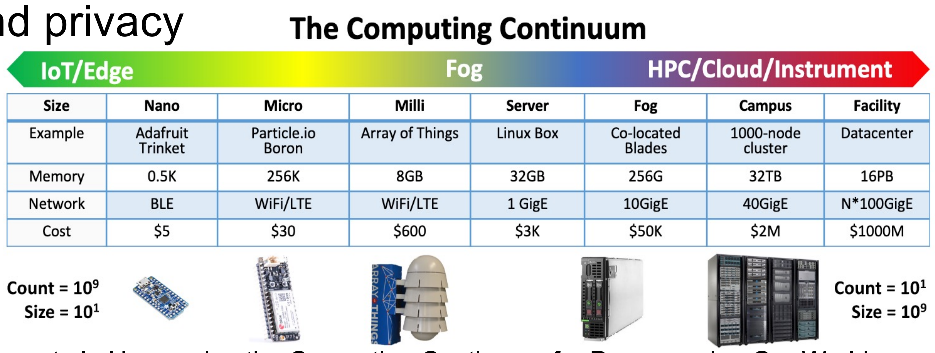
Ultimo aggiornamento 29.08.2023 06:28 | 11242 visible sensors out of 11380 | Measurements by Sensor.Community contributors



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

Computing continuum: challenges

- Infrastructure and application management: interoperability and seamless integration
- Resource heterogeneity (capacity, energy, ...)
- Performance guarantee
- Security and privacy



The Computing Continuum

IoT/Edge — Fog — HPC/Cloud/Instrument

Size	Nano	Micro	Milli	Server	Fog	Campus	Facility
Example	Adafruit Trinket	Particle.io Boron	Array of Things	Linux Box	Co-located Blades	1000-node cluster	Datacenter
Memory	0.5K	256K	8GB	32GB	256G	32TB	16PB
Network	BLE	WiFi/LTE	WiFi/LTE	1 GigE	10GigE	40GigE	N*100GigE
Cost	$5	$30	$600	$3K	$50K	$2M	$1000M

Count = 10^9 Size = 10^1 … Count = 10^1 Size = 10^9

Beckman et al., Harnessing the Computing Continuum for Programming Our World, 2020 https://www.netlib.org/utk/people/JackDongarra/PAPERS/icl-utk-1358-2020.pdf

- Methodologies, programming models, tools and platforms for computing continuum are still in their infancy

Summing up: cloud benefits

IT 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

Business benefits

- No advanced payment
  - Costs scale with use
  - CapEx to OpEx
- Rapid business innovation
- Increase in productivity

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?
  - In Europe: CISPE Code of Conduct
    https://www.codeofconduct.cloud/
  - Be aware of privacy policies
    - "We use the information we collect in existing services to help us develop new ones. For example, understanding how people organized their photos in Picasa, Google's first photos app, helped us design and launch Google Photos." (Google, 2024)

      https://policies.google.com/privacy
    - "Microsoft uses the data we collect to provide you with rich, interactive experiences. (…) We also use the data to operate our business, which includes analyzing our performance, meeting our legal obligations, developing our workforce, and doing research." (Microsoft, 2024)

      https://privacy.microsoft.com/en-gb/privacystatement

Issues for cloud customers

- Network latency
  - ms latency from users and devices to cloud data centers
  - Latency-sensitive apps (e.g., self-driving vehicles, XR and metaverse)? Computing continuum

- Cloud portability
  - Customers ability to move and suitably adapt their apps and data between their own systems and cloud services, and between cloud services of different providers and potentially different cloud deployment models
  - Risks of vendor lock-in
    - E.g., specific features offered by PaaS provider that complicate switching to another provider
    - E.g., difficulty in getting data out and migrate to different provider
  - Towards portability
    - Containers
    - Cloud automation tools

Issues for cloud customers

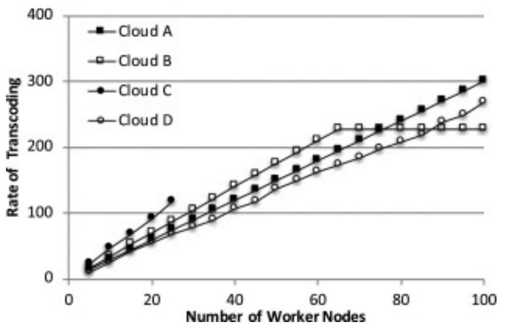
- Cloud interoperability
  - Ability of public cloud services, private cloud services, and cloud service customer's system to understand each other's interfaces, configuration, forms of authentication and authorization, data formats, etc. in order to cooperate and work with each other
  - Standards for 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?
    - Rethink application design for cloud!
  - Is cloud provider able to support real elasticity?
    - *Automatically* provision and de-provision resources on demand as many as required, while maintaining availability and performance

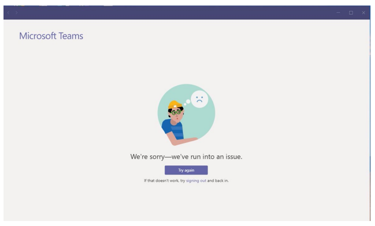
Horizontal scaling in public clouds shows some limit

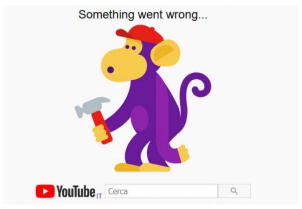


Jiang et al., The Limit of Horizontal Scaling in Public Clouds, ACM TOMPSEC, 2020
https://media.qyjohn.net/blog/2020_Horizontal_Scaling.pdf

Issues for cloud customers

- Cloud outages: some example
  - 2017: AWS S3 went down for 4 hours, taking Slack and several media sites down with it, due to human error https://aws.amazon.com/message/41926
  - 2023: Google cloud services in Europe hit by outage when a data center in Paris 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
  - 2024: AWS outage due to connectivity issues hit Amazon Services and subsidiaries (Whole Foods, Alexa) https://tinyurl.com/bdhpmmw2
  - More examples in 2024 https://tinyurl.com/kky74fmz





Issues for cloud providers

- Increasing energy consumption
  - Energy consumption of data centers: 200 TWh in 2016, 460 TWh in 2022, 1000 TWh in 2026
- Uncertainty in service demands
  - An 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

Summing up

- Cloud computing is:
  – here to stay and one of the most hyped subjects in IT
  – increasingly becoming an integral concept in IT
  – major trend in the way digital services are designed, implemented and delivered
  – key factor for the digitalization of the whole 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://dl.acm.org/doi/pdf/10.1145/1721654.1721672
- The NIST definition of cloud computing https://csrc.nist.gov/pubs/sp/800/145/final
- Cloud SLAs: Present and future http://www.cs.columbia.edu/~salman/publications/baset-sla-osr.pdf
- 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://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=495c97fc8b95492ba6161c86588eaa5b56b63253
- A manifesto for future generation Cloud Computing: Research directions for the next decade https://arxiv.org/pdf/1711.09123.pdf