Cloud Deployment and Service Models

Servicios y Aplicaciones Distribuidas

Cloud Computing Recap

- Shared pool of compute, storage, networking on demand.
- On-demand self-service, broad access, elasticity.
- Measured service: pay for what you use.

Why Service Models Matter

- They define the abstraction level you consume.
- Control vs convenience trade-off changes per model.
- Impacts agility, cost, and who is responsible for what.

IaaS: Definition & Scope

- Virtualized compute, storage, networking primitives.
- You manage OS, runtime, middleware, data, apps.
- Examples: EC2, Azure VMs, Google Compute Engine.

IaaS: Benefits and Challenges

- Benefits: flexibility, compatibility, fine-grained control.
- Challenges: patching, capacity planning, more ops toil.
- Mitigation: golden images, IaC, autoscaling groups.

PaaS: Definition & Scope

- Deploy code; platform manages runtime, scaling, and OS.
- Opinionated buildpacks and service bindings.
- Examples: Heroku, Azure App Service, AWS Elastic Beanstalk.

PaaS: Benefits and Challenges

- Benefits: faster delivery, less undifferentiated heavy lifting.
- Challenges: platform constraints, vendor features vary.
- Mitigation: 12-factor practices, clear SLOs, platform guardrails.

SaaS: Definition & Scope

- Complete application delivered as a service.
- Provider manages everything; you configure and consume.
- Examples: Gmail, Salesforce, GitHub, Slack.

SaaS: Benefits and Challenges

- Benefits: fastest time to value, minimal ops burden.
- Challenges: customization limits, data/control constraints.
- Mitigation: evaluate APIs, export paths, compliance posture.

IaaS vs PaaS vs SaaS (Responsibilities)

- IaaS: you manage OS → app; provider manages hardware.
- PaaS: you manage code & data; provider manages runtime & OS.
- SaaS: you manage config & users; provider runs the app.

Choosing the Right Model

- Drivers: time-to-market, control needs, compliance, skills.
- Workload traits: variability, latency needs, statefulness.
- Evolve per service—mixed models across a portfolio.

Cloud Providers Overview

- Major hyperscalers: AWS, Azure, Google Cloud.
- All offer IaaS/PaaS/SaaS building blocks.
- Differ in strengths, ecosystem, and pricing models.

AWS: Strengths & Focus Areas

- Breadth/depth of services; rich ecosystem.
- Mature IaaS (EC2, VPC), serverless (Lambda), data (S3).
- Marketplace, partners, and global footprint.

Azure: Strengths & Focus Areas

- Enterprise integration with Microsoft stack.
- Hybrid tooling (Azure Arc), strong identity (Entra ID/AAD).
- Windows/.NET first-class, growing Linux/K8s support.

Google Cloud: Strengths & Focus Areas

- Data, analytics, and ML leadership (BigQuery, Vertex AI).
- Innovations in containers and SRE practices.
- High-performance networking.

Multi-Cloud and Portability

- Reasons: resilience, data locality, negotiation leverage.
- Costs: duplicated ops, lowest-common-denominator designs.
- Pragmatic approach: portable apps; non-portable data/analytics.

Public Cloud: Definition & Benefits

- Shared infrastructure operated by a provider.
- Elasticity, global reach, rapid innovation.
- Lower upfront capex; pay-as-you-go.

Public Cloud: Risks & Trade-Offs

- Shared tenancy and noisy-neighbor effects.
- Potential egress and premium managed-service costs.
- Compliance and data residency constraints.

Private Cloud: Definition & Benefits

- Dedicated infrastructure for one organization.
- Control, customization, and data locality.
- Predictable performance and governance.

Private Cloud: Risks & Trade-Offs

- Higher capex and ops burden vs public cloud.
- Slower access to new managed services.
- Risk of under/over-provisioning capacity.

Hybrid Cloud: Definition & Benefits

- Combine public and private across one architecture.
- Burst capacity and place workloads where they fit.
- Data gravity respected; latency-sensitive parts stay close.

Hybrid Cloud: Risks & Trade-Offs

- Complex networking, identity, and policy management.
- Operational duplication across environments.
- Hidden latency and failure modes across links.

Community Cloud (Brief)

- Shared by organizations with common concerns.
- Often compliance-driven (sector-specific).
- Niche, but relevant in regulated domains.

Use Case: Web Applications

- Stateless tiers fit PaaS/serverless; databases managed.
- CDN, WAF, and auto-scaling patterns are common.
- Blue-green/canary releases reduce risk.

Use Case: Big Data & Analytics

- Object storage as data lake; serverless query engines.
- Batch and streaming pipelines (managed services).
- Governance: catalogs, lineage, access controls.

Use Case: AI/ML Workloads

- Managed notebooks, training, and inference endpoints.
- Access to accelerators (GPUs/TPUs) on demand.
- Feature stores and MLOps pipelines.

Use Case: Backup & DR

- Cross-region replication and immutable backups.
- Tiered storage classes to optimize cost.
- Test restores and document RTO/RPO.

Cloud Cost Models

- On-demand, reserved/committed use, and spot/preemptible.
- Per-request/serverless pricing vs provisioned capacity.
- Data egress and managed service premiums matter.

FinOps Basics

- Visibility: tagging, allocation, showback/chargeback.
- Optimization: rightsizing, autoscaling, schedules.
- Governance: budgets, alerts, guardrails.

Shared Responsibility Model

- Provider secures the cloud; you secure what you run.
- Responsibility shifts by service model (IaaS→SaaS).
- Document controls: identity, network, data, app.

Compliance & Data Residency

- Regions and availability zones influence data location.
- Certifications (ISO, SOC) ≠ compliance out-of-the-box.
- Design for audits: logging, retention, access reviews.

Vendor Lock-In & Portability

- Managed services speed delivery but deepen coupling.
- Containers and open standards improve portability.
- Make exit plans: backups, data export formats.

Key Takeaways

- Pick the right abstraction per workload (IaaS/PaaS/SaaS).
- Public, private, and hybrid each trade control vs agility.
- Governance, cost, and security must be intentional.

Closing & Next Session

- Next: Synchronous vs Asynchronous Communication.
- Prepare: API examples, messaging fundamentals.
- Q&A and reading list.