DES424 Lecture Summary

Cloud-based Application Development

SIIT DE-ASD Y4T1/2022 – By Paphana Yiwsiw (@waterthatfrozen)

Contents

Lecture 1 –Introduction

Software development and engineering

- Why is it difficult? → Difficult problem/solution domain, development process. Discrete and Flexibility of software. (Discrete systems can have hidden surprise)
- Definition of Software engineering:
  "A collection of techniques, methodologies, and tools that help with the production of **high-quality in-budget on-time flexible** software system"
  Software engineering is used to create system based on the solution for a problem, usually challenged by dealing with the complexity and changes.
- Software engineering "project" is to develop software system
  A project is composed of many activities. Each activity consists of many tasks.
  Work product is produced by a task, it can be either system, model, or document.
  A task consumes many resources, such as human, time, and equipment.
- Software engineering participants and roles includes:
  - Client → Provide high-level requirements, define the project scope
  - User → The one who use the system to satisfy their tasks.
  - Manager → Organizing works, managing resources, etc.
  - Developer → Construct the system based on the specifications, design, etc.
  - Others → Human factor specialist, technical writer.
- Work product can be deliverable(Specification/Operation manual) or internal work product (status report/test manual)

Software Life Cycle Activities

- Set of activities and relationships to support the development of software system
- **Requirement Elicitation**(Use case model) → **Analysis**(Application domain objects) → **System Design**(Subsystems) → **Object Design**(Solution domain objects) → **Implementation**(Source code) → **Testing**(Test case)

Types of Requirements

- **Functional requirements:** Interaction between the system and environment independently from the implementation
- **Nonfunctional requirements:** Aspects not directly related to functional behavior.
  - **Quality requirements**
    - **Usability:** Ease of use to perform function in the system, must be measurable.
    - **Reliability/Robustness:** Maintain a function ability of a system, foolproof from users and able to handle environment changes.
    - **Performance:**
      - **Availability:** Ratio of expected uptime to expected uptime and downtime.
      - **Others:** Response time, scalability, throughput.
    - **Supportability:** Adaptability, maintainability
  - In the requirements analysis document template, section of nonfunctional requirements consists of:
    - UI and human factors: type of user, training, easy to learn, foolproof, I/O
    - Documentation: kind of documentation, audience
    - Hardware consideration: which to be use, characteristic of target hardware
    - Performance characteristics: speed, throughput, response time, data size
    - Error handling and extreme conditions: response to error and extreme conditions
    - System interfacing: Input and output from outside, restriction on the format

        - Quality issues: reliability requirement, trap fault, acceptable downtime, portability
        - System modifications: parts to be modified, what kind to be expected?
        - Physical environments: target equipment, locations, ordinary environments.
        - Security issues: access to data or controlled system, physical security.
        - Resources and management issues: back up frequency, staff for installation and maintenance.
- **Constraints:** Imposed or restricted by the client or the environment
  (can be called pseudo-requirements)
    - Implementation / Interface / Operation / Packaging / Legal
    - Delivery constraints: when to deliver the product
    - Organizational constraints: who and how we organize people
    - Implementation constraints: what languages or technologies to be used
    - Target platform constraints: what platform it will operate.

Reusability

- A good software design involves solving a specific problem but general enough to solve future problems. Most people tend to reuse solution that worked for them in the past.
- "Design the software to be reusable across application domains and designs" → Achievable by using **architectural styles, frameworks, and design patterns.**
- Design pattern
    - A small set of classes that provide a template solution to recurring design problem
- Framework
    - A large set of classes that collaborate to carry out a set of responsibilities in the application domain
- Both design pattern and framework provide architectural guidance in the design phase and foundation of software components.
- Popular frameworks:
  Git/GitHub/SVN for Version control, Jenkins for CI/CD, Selenium & Robot for automated testing

Lecture 2 – Cloud Computing

Cloud Computing Definition

- **Physical**: Global deployment of data centres connected by fast networking, designed for scalability and robustness
- **Logical**: A collection of tools and platforms that scale well. The platform matters the most. As a developer, they allow you to extend and customize to create application as a personality over their capabilities.
- **Conceptual**: Set of scalable idea, concepts, and technologies
- It is a new approach that reduces IT complexity by leveraging the efficient pooling of on-demand self-managed virtual infrastructure, consumed as a service.
- Internet-based computing, whereby shared resources, software, and information are provided to computers and other devices on demand.
- Specialized form of distributed computing that promote utilization models for remotely scalable and measured resources.

Business driver is the reason for choosing cloud

- Capacity planning; planned vs actual capacity. It may underestimate or overestimate.
- Cost reduction: IT Infrastructure acquisition/maintenance
- Organizational overhead
    - Person to maintain physical infrastructure
    - System upgrade, utility bill, physical security access, etc.
- Organizational agility
    - Ability to adapt and evolve infrastructure to handle changes from business factors.
    - Funding constraints lead to insufficient IT infrastructure. However, Cloud computing allows IT resources to scale with lower financial support.

Computing Technology Innovation

- Cluster computing
    - A group of independent IT resources interconnected as a single system
    - Configured servers with homogeneous hardware (same RAM, CPU, HDDs) and software
    - Enables **warm replica** servers, duplicate key infrastructure to provide hardware failover to ensure high availability. Keep all components running by emphasizing redundancy.
- Grid computing
    - A platform in which computing resources are organized into one or more logical pools
    - Collectively coordinated to provide high-performance distributed grid "Super virtual computer"
- Virtualization
    - Technology platform used for creation of virtual instances of IT resources
    - Allows physical IT resources to provide multiple virtual images, thus underlying processing capabilities can be shared by multiple users.
    - Simulated hardware via software, VM, Virtual Network , Virtual Disk
- Others: network/disk technology, hardware cost and specification

Key Terminology

- On premise Infrastructure: local server not configured as cloud., infrastructure in office.
- Cloud provider: corporation or organization responsible for maintaining cloud

- Cloud consumer: cloud services user
- Cloud services: any IT resource that is made remotely accessible via a cloud.
    - Can be either Simple web with interface or remote access point for admin tools
- Vertical scaling: increase(up) or decrease(down) resources of a single virtual server
    - Configure different resources: CPU cores, RAM, HDD/SDD
    - May require VM migration if physical host are exceeded
- Horizontal scaling: increase(out) or decrease(in) number of virtual servers
- Service level agreements (SLAs)
    - Established expectation for uptime, security, availability, reliability, and performance

Essential Characteristics of Cloud Computing

- On-demand service
    - Unilaterally provision computing capabilities as needed automatically without requiring human interaction with each service provider. Automated through software interface.
- Broad network access
    - Available over the network and accessed through standard mechanism
- Resource pooling
    - Resources are pooled to serve multiple consumers, dynamically assigned and reassigned to the demand.
- Rapid elasticity
    - Automated ability of a cloud to transparently scale IT resources, as required in response to runtime conditions or as pre-determined by the cloud consumer or provider
    - Expanded or released automatically
- Measured services
    - Ability of a cloud platform to keep track of the usage of its IT resources, usually by cloud consumer.
    - Metering concept where resource usage can be monitored, controlled, and reported

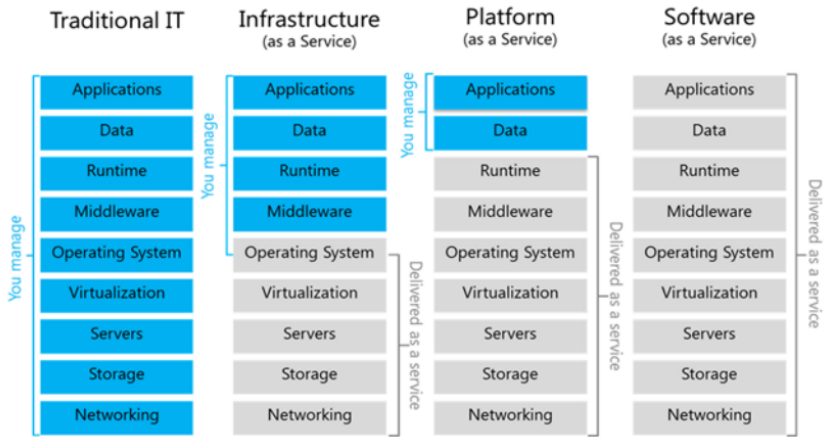
Advantages and Disadvantages

- Advantages: cost savings, ease to install and maintain, increase storage, highly automated, flexible, better reliability, shared resources, back up and restore.
- Disadvantages: data security and privacy, compete for bandwidth, unavailability of service, high cost if not planned carefully, knowledge and integration, long term stability.

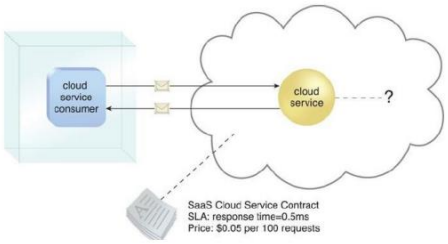
Security Challenges

- Cloud security challenges include:
    - trusting security model of vendors,
    - multi-tenancy,
    - data ownership/privacy issues,
    - QoS guarantees,
    - attracting to hackers,
    - obtaining support from vendors for investigation in security.

Cloud Service Models

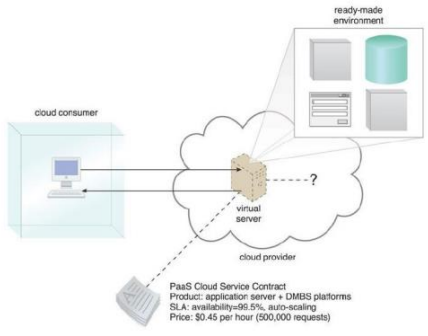


SaaS: Software as a Service



- Provide application over a network to end-users
- A shared cloud service is a software program and **available as product** or generic utility
- You can select software or services.
- Benefits: **Lower costs**, **no infra required**, seamless upgrade, guaranteed performance, automated backups, easy data recovery, secure, **high adoption**.
- Characteristics: Multi-tenancy, **On-demand** software, **Open integration** protocols, social network integration.
- Adoption: Mostly from **users and SMEs**, not so much from governments and large organizations.
- Examples: Google Suites, Salesforce, Office 365, Facebook, etc.

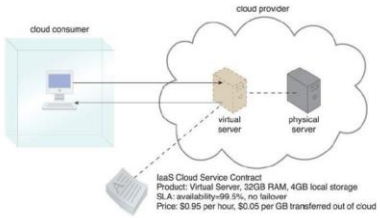
PaaS: Platform as a Service



- **Develop** application, OS, deployment framework such as database or web server to developer.
- "**Ready-to-use**" environment, mostly consist of deployed and configured IT resources
- No need to care how to install and configure the platform, consumers can select database platform, web platform, message queue platform, etc.
- Benefits: **Lower upfront** and ops costs, **no IT infra** costs, Improved **scalability**, higher performance, secured access, **quick** and easy development, **seamless integration**.

- Characteristics: Multi-tenancy, Open integration protocols, App dev tools and **SDKs**, Analytics.
- Adoption: Mostly **Large organizations**, not a lot from SMEs and Governments, low from users.
- Examples: Google App Engine, Microsoft Azure, GitHub, Database, Web Server, etc.

IaaS: Infrastructure as a Service



- **Provision** on-demand VM, storage, and network **resources** to IT admin and network architects.
- You can select compute set, storage, network (public IP), pre-installed OS.
- Benefits: **shift focus** from IT management to core activities, **no IT infra** management costs, **pay-per-use** price, guaranteed performance, **scaling** dynamically, secure access, **enterprise grade** infrastructure
- Characteristics: Multi-tenancy, **virtualized** hardware, management and monitoring tools, **disaster recovery**.
- Adoption: Mostly **large organizations and governments**, not a lot from SMEs, low from users
- Examples: Azure VM, Amazon elastic compute cloud: EC2

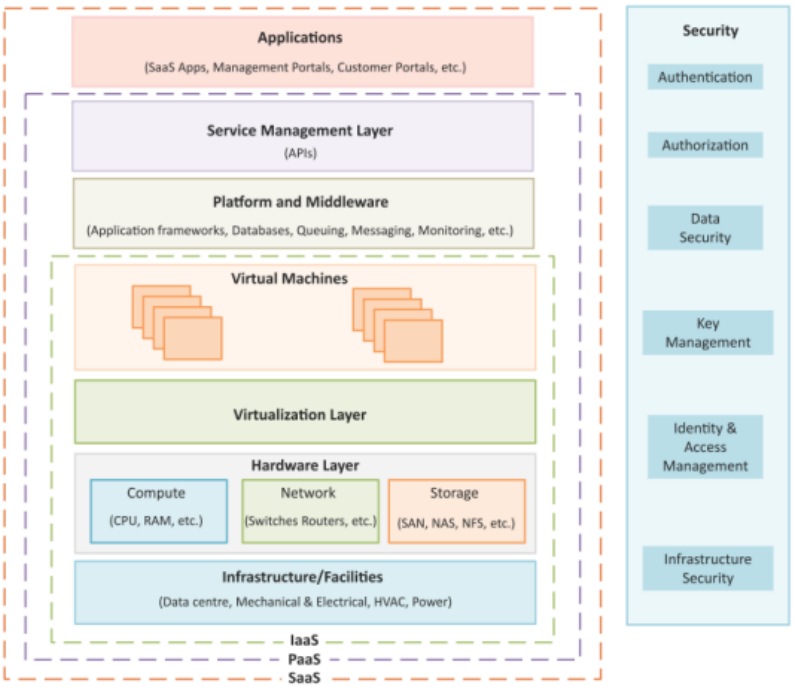
Reality

A combination; IaaS: Web Server, Web App, API / PaaS: MS SQL Server / SaaS: Authentication Service

Cloud Deployment Model

- Public Cloud: public usage, convenient, scalable, reliable, cost-effective, third-party
- Private Cloud: private owned, customizable, reliable, scalable, on-premises
- Hybrid Cloud: Highly flexible, dataset segmentation, scalable, Public + Private, Need expert
- Community Cloud: Crowdsourcing, limited for access by a group, cross-organization

Cloud Services & Platforms



- Compute Services
    - Dynamically scalable compute capacity in the cloud
    - VM can created from standard images from provider or user-customized
    - Accessible from the web consoles with GUI for managing and monitoring
    - Provided APIs to allow developers to manage services
- Storage Services
    - Allow storage and retrieval of any amount of data at anywhere and anytime with scalability, replication, access policies, encryption, and encryption features.
- Database Services
    - Allow to set up and operate non-relational and relational databases with scalability, reliability, performance, and security. Such as Azure Cosmos DB or Azure DB for MySQL
- Application Services
    - Application runtimes and frameworks, allow dev to develop and host app in the cloud
    - Services: Queuing, Email, Notification (Push mobile noti.), Media Services (Transcoder)
- Content Delivery Services
    - Distributed system of servers located in many locations to serve content to end-users with high availability and high performance, serving static content e.g., text, images, and steaming media.
    - Many edge locations deployed in multiple locations, often over multiple backbones.
    - Request are served by a CDN are directed to the nearest edge location
    - Example: Amazon CloudFront, Azure CDN
- Analytics Services
    - Analyzing massive datasets stored in the cloud either in cloud storage or databases using programming models such as Amazon Elastic MapReduce
- Deployment & Management Services
    - Easily deploy and manage application in the cloud.
    - Automatically handle deployment task such as capacity provisioning, load balancing, auto-scaling, and health monitoring.
- Identity & Access Management Services
    - Managing authentication and authorization for securely access cloud resources.
    - IDAM services for managing user identifiers, permissions, credentials, and access key.
    - Examples: Amazon IAM, Azure IAM as RBAC (role-based access control)

Design Considerations for Cloud Applications

- Business Need: goals, requirements, performance objectives, **target environment** (deployment model, cloud services and providers)
- Investment Costs: recurring costs and initial costs
- Technical issues: **solution architecture**, **subsystem decompositions, data management, boundary conditions,** SLAs, requirements of security, governance, interoperability, portability, regulatory compliance, approach for migration, and maintenance and upgrade.

In the old approach, you built web page in location of the web server, static routing with no load balancing. The application just used **3-tier Application Architecture.**
With the new approach, you implemented cloud resources to become **3-tier application architecture with cloud resources**, user computers routed the request to the nearest among all data centers, web page built at that server, there is also a backup route if the nearest one failed, the databases are split into smaller and highly parallel services, so web server become simpler and done less work.

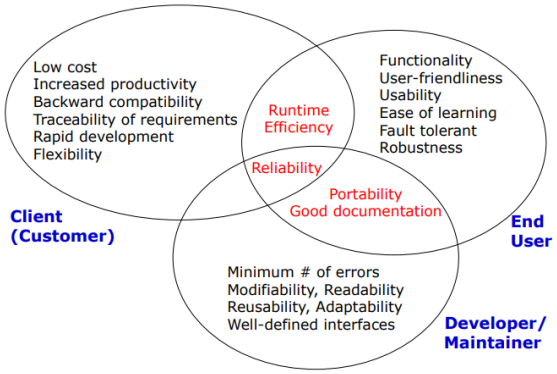
Lecture 3 – Cloud Computing and Design Concepts

Cloud Providers

- Microsoft Azure: PaaS, enterprise-level on-demand, use Azure API to work with infrastructure offered, significant feature: web role, worker role, blob storage, etc. Provide software for build up local azure cloud on our infrastructure.
- GCP: Suites of cloud computing services that run on same infrastructure of Google uses for its products. Provide a set of management tools, series of modular cloud services including computing, data storage, data analytics, and ML.
- AWS: EC2 large complex web service, provide API for instantiating computing instances with any of the operating systems supported. Facilitate computations through Amazon Machine Images, excellent distribution, load balancing, cloud monitoring tools.

Design Considerations

Stakeholders' focus



Everyone focuses about reliability, users and developers wants good documentations, customers and end users wants runtime efficiency.

Subsystem Decompositions

Collection of classes, associations, operations, events, and constraints that are closely interrelated with each other. Initial decomposition based on functional requirements. For example, user management, user directory, user interface, advertisement, statistics, etc.

Data Management

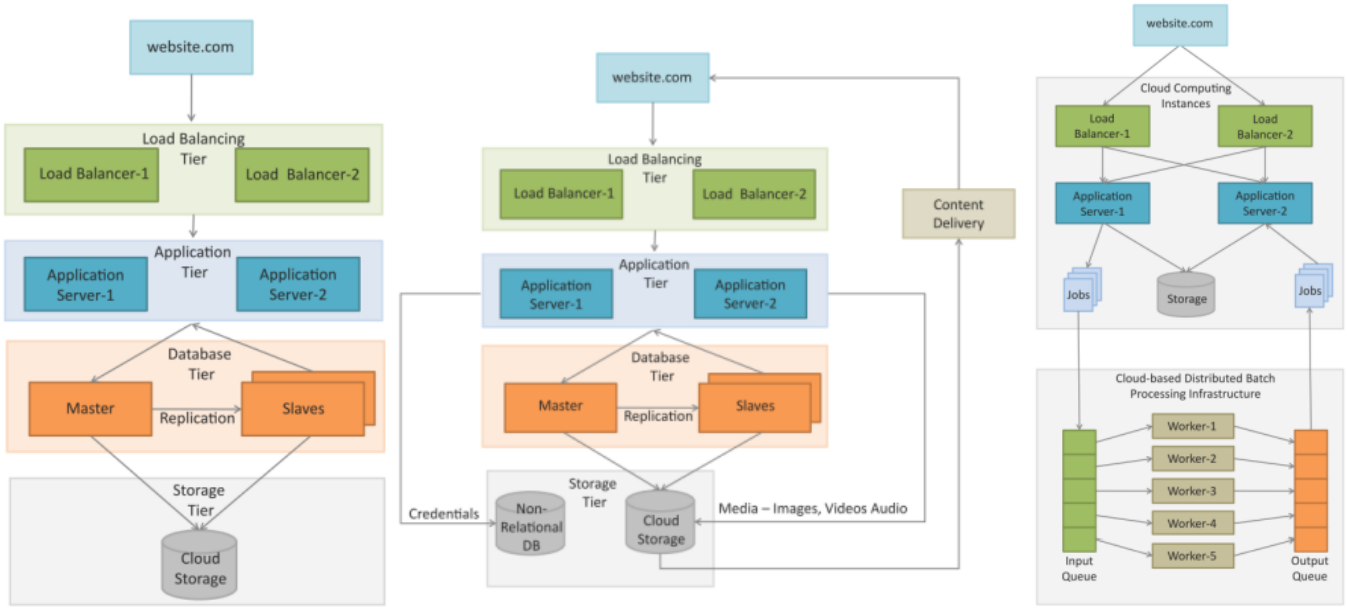
- Some objects in the system model must be persistent: attribute's value have a lifetime longer than a single execution. It can be realized with the file system (single writer, multiple reader) or the database (concurrent writers and readers).
- Should data be distributed (location transparency)? Active/standby, load balance, branch

Boundary Conditions

- **Initialization**
  - System from non-initialize to steady state
  - What data need to be accessed at start-up time, services to registered, UI behavior at the start-up time?
- **Termination**
  - Clean up resources and other systems are notified upon the termination.
  - Is single subsystem allowed to terminate, notify if one subsystem terminate?
  - How are updates communicated to the database?

- **Failure**
    - Possible failures of system: Bugs, errors, external problems
    - How does the system behave when a node or comm. link fails, recover from failure?

Reference Architectures



Left: Normal reference architecture, Center: Architecture with CDN included, Right: Architecture of Analytics apps.

- **Load balancing tier**: one or more of load balancers
- **Application tier**: configure to auto-scaling, it can be triggered when the recorded values for any resource's metrics e.g., CPU usage, went above specific thresholds.
- **Database tier**: Include single master database and many slave instances. Master serves all write requests while read requests served to slave nodes. It improves throughput because most applications have higher read requests than write.

If CDN is included, both relational and non-relational database are shown, CDN is used for media delivery which consists of many edge locations in the global network, used to speed up the delivery of static content such as images and videos.

In case of analytics app, it is a computationally intensive applications such as data analytics, transcoding, etc. It consists of cloud-based distributed batch processing framework such as Hadoop for big data analytics. Data analysis jobs are submitted to analytics tier from the application servers, and queued for execution, then presented analyzed data from the application server after completed.

Lecture 4 – Cloud Infrastructure

How can we build Cloud?

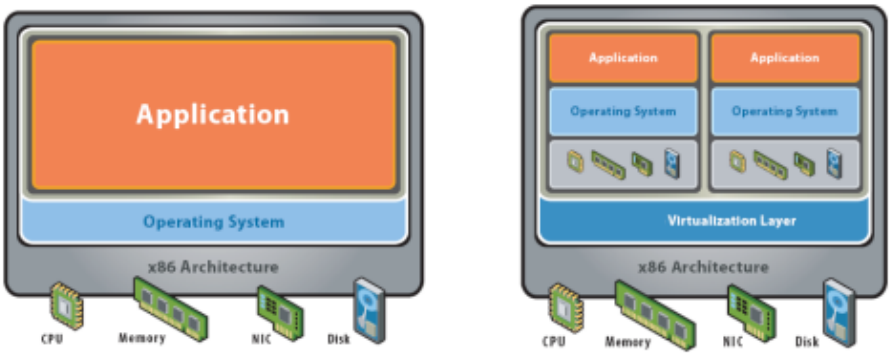
- **Hardware**: enterprise grade CPU, memory, storage. Supporting VM, clusters, and resource pools
    - Must be able to run 24/7.
    - Type of servers: tower (mostly used in-house, no use in data center), rack (mostly used in data center, many subtypes of it: CPU, storage, GPU)
    - Check the specification carefully as most servers use CPU from Intel Xeon family, check the number of cores, threads, multithreading capacity.
    - Memory of servers is also important, usually used 16-32GB each slot but it must also be compatible, disk such as HDD/SSD/NVME and set to raid 1, 5, or 10, network interface such as 2.5 or 10GB port for networking and fiber optic cable to support that speed, and GPU between on-board or RTX but may need to think about power consumption.
    - For storage, SSD for OS, normal HDD for storage, no NVME usage due to its high cost despite its huge read and write speed.
- **Location**: places to put data centers and servers
    - Thousands of closely coupled hosts with power sharing and higher efficiency in shared IT resources usage (less dupe) and improve accessibility and organization.
    - Key components: virtualized and physical server resources, standardized modular hardware, automation support (server provision/config/monitor without supervision)
    - Components of data center
        - Remote operation
        - High availability support: **Redundant everything** even power supply, environment control system, communication link, cabling, *warm replica hardware*
        - Both physical and logical secure design
        - Rackmount server
        - Hard disk arrays storage (SAN: storage area network → disk array with multiple servers and a dedicated network)
        - NAS: Network attached storage: inexpensive single node with collection of disks and shared filesystems for NFS
        - Network hardware: backbone routers (WAN→LAN), firewalls, VPN gateways, managed switches, and routers
    - Tier of Data Centers
        - Tier 1: Basic → Single paths for power and cooling, non-redundant component, optional raised floor, subject to disruption of hardware operations.
        - Tier 2: Redundant Components → redundant components, mandatory raised floor, power path may disrupt hardware operations.
        - Tier 3: Concurrently Maintainable → Multiple paths for power and cooling, maintenance without disruption
        - Tier 4: Fault-Tolerant → Fault-tolerant components, planned activities don't affect critical load, unplanned worst-case failure during maintenance can be sustained without disruption of hardware operations, up to 99.995% of average availability.
    - "**Co**location Services" → Put your server in data center
- **Network**: Tier 1-ISP → Extremely fast internet connection for everyone to access it.
    - **ISP**: Internet service provider, company that provides connection and support to access the internet, anyone who wants to connect to the internet must use an ISP, varies in size and terms of their service area, might be limited depending on the geographic location.

- ISP services includes internet, email, hosting, colocation, cloud, access network.
- **ISP Category:**
  - Tier 1/Backbone
    - National or multi-national company control routing, own significant pieces of backbone, settlement-free interconnection
  - Tier 2/National
    - Tier-2 ISP buy bandwidth/capacity and routing services from Tier-1 and run point-of-presence (POP) over the country.
    - Purchase IP transit to reach some portion of the internet from Tier-1
  - Tier 3/Local
    - Operate same as tier-2 ISP but smaller geographical area
    - Purchase IP transit to reach internet from Tier-2

- **Software**: virtualization tools/software such as VMware and Proxmox
  - **Convert physical to a virtual IT resource.**
  - Virtualization support hardware independence, server consolidation, resource replication, resource pooling, elastic scalability
  - Virtual servers can be OS-based or hardware-based virtualization

Virtualization

- Virtualization technology



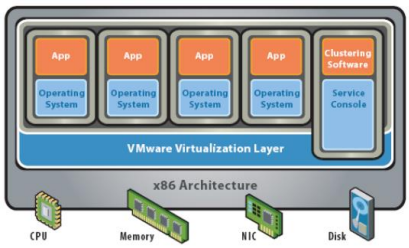
  - Before(Left): 1 OS per machine, tightly coupled software and hardware, underutilized resources, inflexible and costly infrastructure.
  - After(Right): hardware-independence OS, VM can be provision to any system, manage OS and application as single unit by encapsulate them into VM
  - Required 64-bit multi-core architecture,
- Why use virtualization
  - Workloads consolidation to reduce hardware cost
  - Single consolidation view and management of VMs
  - Portability of VMs, can migrate from one to another without shutting down
  - Isolation VMs are protected from each other and from hardware resources
  - Can used for testing, training, software development; run many OSs simultaneously
  - ISP hosts/cloud computer providers can divide 1 physical machine to different customers
- Concerns on virtualization
  - Overhead with too many instances and heavy abstraction can result in degradation of performance. Too many of them can lead to hidden resource utilization and waste
  - Performance: impacted by the weight of abstraction and virtualization overhead
  - Security: User A and B data should be always separate.

VMs as Cloud Abstraction

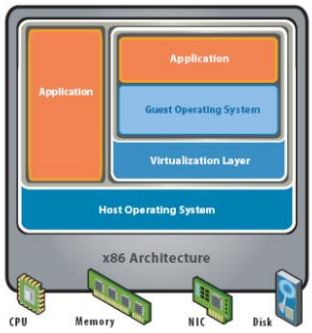
- VMs is an original IaaS cloud abstraction
- OS and Application Containers as CaaS: Container as a Service
    - OS container: **mimics full OS instance**, replacement of VM, heavier, run hundreds of processes like VMs.
    - Application container: docker provide package dependencies to easily transport and **run an application anywhere**, run just a few processes inside containers.
- Micro VMs FaaS: Function as a Service /CaaS → Alternative lighter weight to full VM

Hypervisor

- VMM: Virtual Machine Monitor, used to create and run VMS, allow one host to support multiple guest VMs by sharing its resources virtually, such as memory and processing.
- 2 Types of hypervisors:
    - Type 1: Bare-metal approach



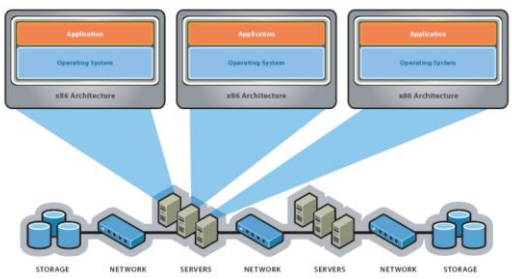
        - No host OS, VMs run on top of type 1 hypervisor directly on a hardware platform
        - No wasted resources for host OS
        - Achieved higher virtualization efficiency
        - Ex. VMware ESXi Server
    - Type 2: Hosted approach



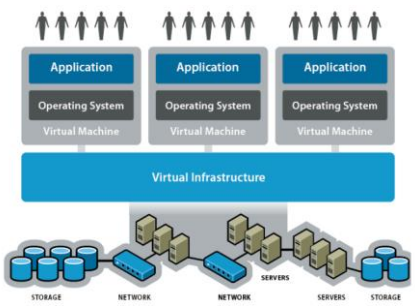
        - Host OS run virtualization software, and unmodified guest OSs run separately on its VMs and isolated from each other
        - Virtualization software is a type 2 hypervisor
        - Additional resources required for host OS, higher overhead
        - Example: Microsoft Virtual PC, VMware workstation

Traditional and Virtual Infrastructure

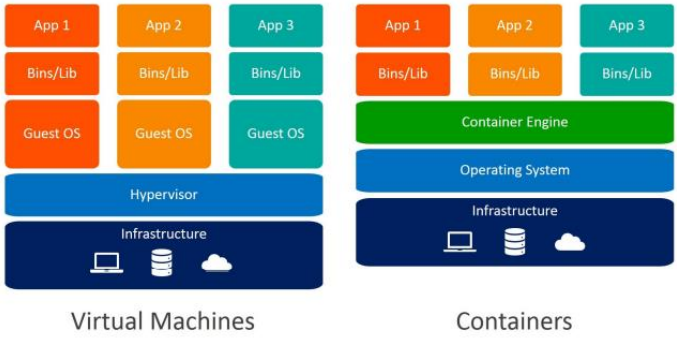
Traditional



/ Virtual



Container



- Products of OS virtualization
- Lightweight virtual environment that groups and isolates a set of processes and resources such as memory, CPU, and disk from the host and other containers.
- Isolation guarantee that any process inside container cannot see any outside the container

Proxmox

- A tool/software for hypervisor
- Proxmox VE: Virtual environment is an open-source software server for virtualization management, Type 2 hypervisor that can run OSs (it can also be type 1)
- Features: server virtualization, kernel-based VM, container-based virtualization, web-based management UI, live/online migration, HA manager, bridged networking, flexible storage options, distributed firewall.
- Proxmox Cluster: Combining multiple servers into a cluster with central management

Extra: JIRA

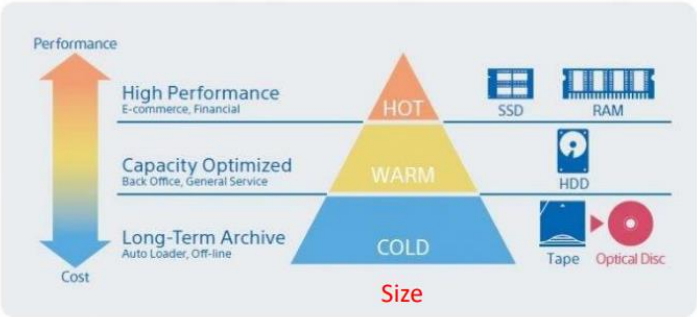
- JIRA is used for project management and tracking
- 2 Project types: team-managed and company-managed
- Epic is the large body of work that can be broken down into smaller stories called issues, it can also have tasks to do inside. (Think of Epic like a feature of application and tasks like requirement of that feature)

Lecture 5 – Cloud Storage

Data characteristics in Cloud Storage

- Physical: size, regulation, ownership, sensitivity
- Performance:
    - Real-time: sub-seconds response time, half-second response time for websites.
    - Near real-time: within 1-2 seconds, perceived as real-time
    - Delayed time: a few seconds to batch time frames of daily, weekly, monthly, and so on.
    - Design patterns for high-volume fast-performing dataset
        - Caching layer, reduce size of dataset by hashing or binary representation, separate database to read-only and write-only, archive aging data, de-normalize datasets.
- Volatility: frequency of data changes
    - Static data:
        - event-driven, chronological order occurrence, write-once read-many
        - stored over longer periods, consume terabytes of data'
        - Ex. Web logs, stock trading, POS purchases
    - Dynamic data:
        - changed frequently, write-many read-many
        - used normalized relational database management system
        - must protect the integrity of the data
- Volume: amount of data that system must maintain and process, both online and offline
- Transaction boundary: a unit of work
- Retention period: period of keeping the data

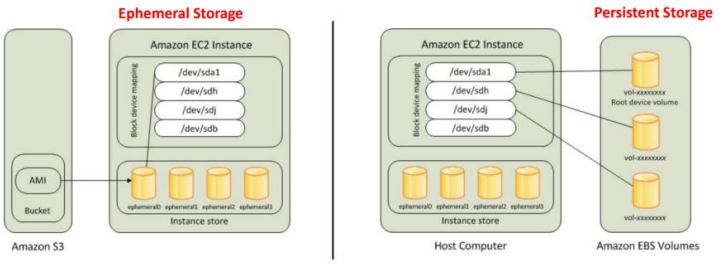
Hot/Warm/Cold Data



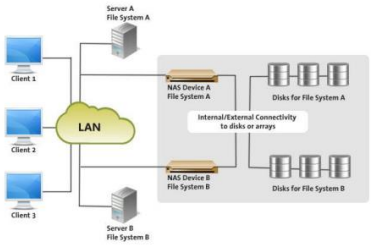
- Hot: consider high performance, frequently accessed, such as financial data, use SSD or RAM.
- Warm: capacity optimized, less frequently accessed, such as general service, use HDD.
- Cold: long-term archive, auto loaded, off-line, rarely accessed, use optic disc???
- Data size: cold data used a lot of storage space, while hot use a little
- Performance-cost trade-off: Hot data is the best one in performance, cold is best in cost.

Cloud Storage

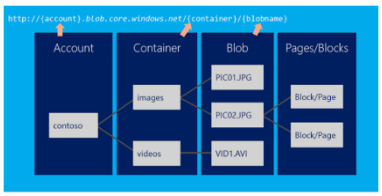
- Storage design specifically for cloud-based provisioning. Instance can be virtualized, provide fixed-increment capacity allocation to support pay-per-use mechanism, exposed remote access via cloud storage services.
- Types of cloud storage: Ephemeral and Persistent



- **Ephemeral Storage**
  - Volatile <u>temporary</u> storage attached to your instances that only available during the running lifetime of that instance, physically connected to the hardware (data can be loss if disk fails, instance stops/hibernates/terminates)
  - Fast read and write speed, used in caching and image processing.
- **Persistent Storage**
  - Always available regardless the state of a running instance, outlive any resources
  - Attached storage
  - 3 Types offered in cloud environments: block, file, and object storage.
    - Block storage
      - Integrated centralized storage into servers as local HDD managed by OS to enable access to this storage via local filesystem
      - The data stored without any data format or type
      - Accessed over the network as SAN using iSCSI protocol: Internet Small Computer System Interface → protocol-based storage networking for linking data storage facilities.
    - File storage

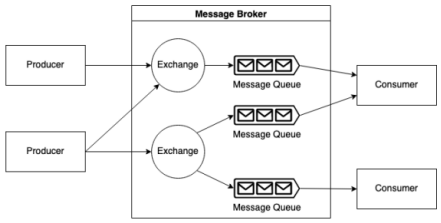


      - Operate at OS user level, managed by filesystem
      - Managed by hierarchy of files and folders
      - Filename, type, date, and other metadata associated with them
      - Accessed over the network as NAS: Network Attached Storage using NFS: Network File System protocol, SMB: Service Message Block, CIFS: Common Internet File System
      - Used in the application of file sharing services, needed the permission control function of users, such as store, retrieve, and share. For file sharing, access by multiple hosts simultaneously.
    - Object Storage



      - Operate at application level, data is stored as objects, each object contains object ID, data, and metadata.
      - Accessed over the network using REST APIs over HTTP protocol
      - Benefits:
        - Greater data analytics; driven by metadata
        - Infinite scalability: no limit, can add data forever
        - Faster data retrieval: due to categorization structure and lack of folder hierarchy
        - Reduction in cost: less costly to store all data due to scale-out nature of object storage
        - Optimization of resources: fewer limitation due to no file hierarchy and customizable metadata.

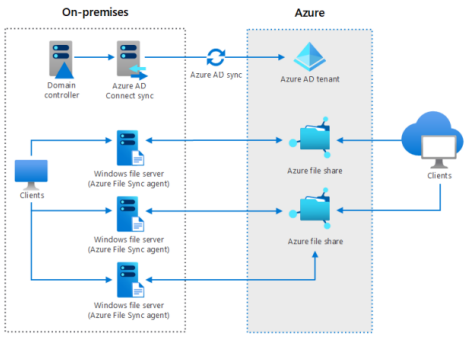
- Additional types of cloud storage
    - Database storage
        - Data is structured according to a schema, enforced during data manipulation, and enable expressive queries of handled data
        - Example: MS SQL, Oracle, MySQL, PostgreSQL, DB2
        - VM need to connect to database using a service endpoint, but need firewall to allow that connection
    - Message/Queue Storage



        - Form of asynchronous service-to-service communication, hiding complexity resulting from addressing, routing, or data formats from communication partners to make interaction robust and flexible. It can be 1-to-many (publish-and-subscribe) or 1-to-1 (point-to-point)
        - One station, one queue: server send command via queue which maintain order, the station sync to server via sync queue

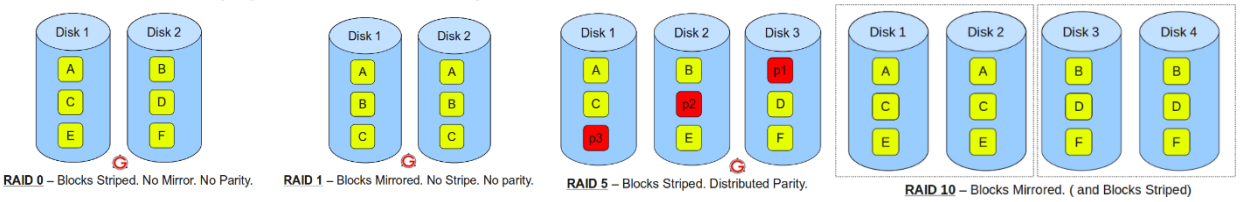
Storage Services

- AWS EC2; Amazon EC2 offers as follows:
    - Amazon EBS: Elastic Block Store
        - Majority of the time, EBS volumes are the recommended because it provides persistent block-level storage
        - Keeping backups copy by creating a snapshot of the volume
    - Amazon EC2 Instance Store
    - Amazon S3: Simple Storage Service
        - CRUD objects up to 5TB each, unlimited number of objects with a unique, stored in bucket and retrieved via a unique developer-assigned key.
        - A bucket can be stored in one of many regions, optimized for latency, reduce cost, or address regulatory requirements.
        - Ensuring that data kept securely from unauthorized access by using authentication mechanisms, which object can be made private or public and giving permission to specific users.
        - Can be used for content storage and distribution, data analysis storage, backup, archive, and disaster recovery.
- Azure File: Share data
    -



RAID Technology

- RAID: Redundant Array of Independent Disks; data storage virtualization technology that combines multiple physical disk drive components into 1 or more logical units for the purpose of **data redundancy, performance improvement,** or both.



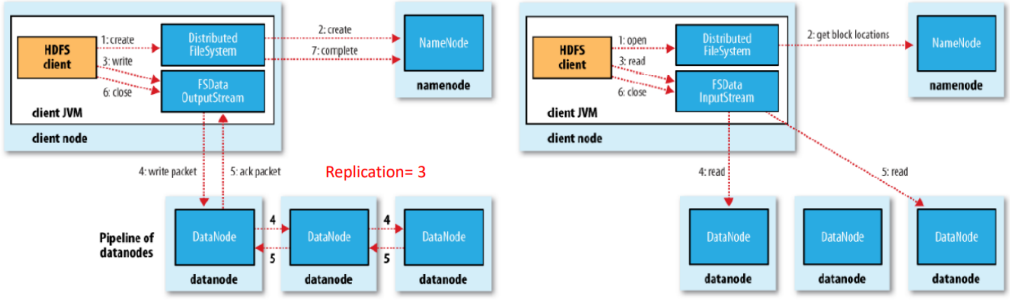
  - RAID 0: Block striped, no mirror, no parity; Don't use for any critical system.
  - RAID 1: Block mirrored, no stripe, no parity; Excellent redundancy.
  - RAID 5: Block striped, distributed parity; Uses disk striping with parity as redundant array of independent disk configuration; Use this in the database with heavy-read operations and cost-effective solution for both performance and redundancy.
  - RAID 10: Strip + Mirror, technically RAID 1+0; Best option to any critical mission
- RAID Connection
  - Hardware RAID: customized processing system, different controllers i.e. RAID cards to manage RAID design independently from the OS, recommended to do so.
  - Software RAID: uses processing power of computer OS in which RAID disks are installed.
- RAID level comparison

RAID Level Comparison

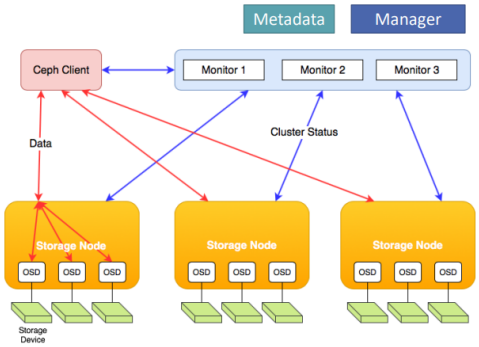
Features	RAID 0	RAID 1	RAID 5	RAID 6	RAID 10	RAID 50	RAID 60
Minimum number of drives	2	2	3	4	4	6	8
Fault tolerance	None	Single-drive failure	Single-drive failure	Two-drive failure	Up to one disk failure in each sub-array	Up to one disk failure in each sub-array	Up to two disk failure in each sub-array
Read performance	High	Medium	Low	Low	High	High	High
Write Performance	High	Medium	Low	Low	Medium	Medium	Medium
Capacity utilization	100%	50%	67% – 94%	50% – 88%	50%	67% - 94%	50% - 88%
Typical applications	High end workstations, data logging, real-time rendering, very transitory data	Operating systems, transaction databases	Data warehousing, web serving, archiving	Data archive, backup to disk, high availability solutions, servers with large capacity requirements	Fast databases, file servers, application servers	Large databases, file servers, application servers	Data archive, backup to disk, high availability solutions, servers with large capacity requirements

Distributed Storage

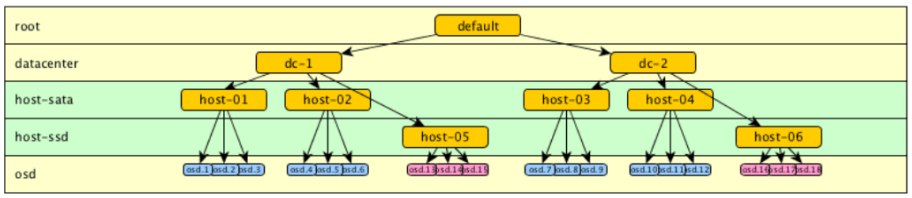
- Infrastructure that can split data across multiple physical servers, mostly more than one center.
- Takes a cluster of storage units form and a mechanism for data synchronization and coordination between cluster nodes.
- **HDFS**: Hadoop Distributed File System
  - Distributed file system designed to run on commodity hardware, used to scale a single APACHE Hadoop cluster to hundreds of them.
  - Data: file split to multiple same-size block and replicate to multiple nodes.
  - Write/Read:



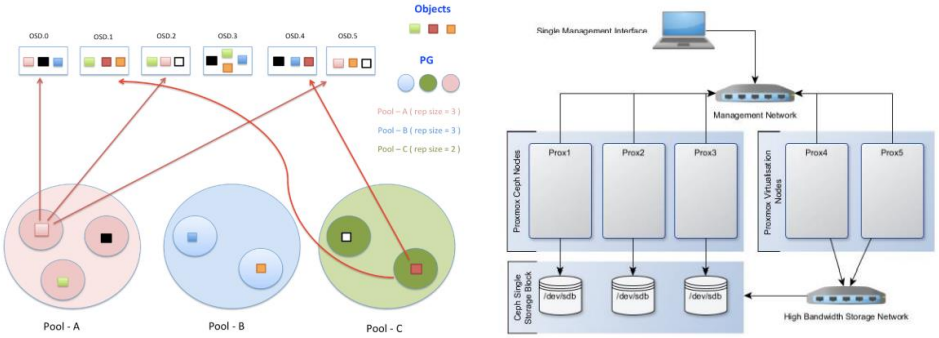
- **Ceph**
  - Open-source software-defined storage platform that implements object storage on a single distributed computer cluster and provide 3-in-1 block, file, object storage
  - Commodity hardware, large scale (up to 10K nodes), communicate to each other to replicate and redistribute data dynamically.
  - Interaction and Members



    - Ceph OSD: checks its own state and the state of other OSDs and report back to monitors. One per disk, serve stored object to client.
    - Ceph Monitor: maintains master copy of cluster map, ensure high availability, storage cluster client retrieve a copy of cluster map from Ceph Monitor
    - Ceph Metadata (MDS): manage file metadata when CephFS is used to provide file services.
  - CRUSH Map: hierarchy that describes physical topology of the cluster and set of rules defining data placement policy.
    - The hierarchy: devices at leaves, physical features or groupings as internal nodes.



  - Placement Group (Pool), Replication / Storage



Software-defined Storage

- TrueNAS: Core
  - Open-source enterprise storage array with features needed by your business application and available and performance.
  - Unified SAN and NAS in one place and provide variety of services and protocols on top of file system that guarantee data integrity at every step.
  - Feature such as Pool manager with RAID feature