

RIASSUNTO DI SISTEMI CENTRALI

(Prof. Messsina F.)

A.A. 2023-24

Modelli Elaborativi

Sistema Centrale (Mainframe)

Calcolatore dotato di **elevata capacità elaborativa**, al quale accedono un **alto numero di utenti** contemporaneamente ed in grado di gestire **grandi volumi di dati**.

Assolvono **3 funzioni fondamentali**:

- Database Server
- Application Server
- Presentation Server

Svolgono principalmente **due tipi di lavoro**:

- **Batch Job**: Applicazioni che girano senza interazione utente gestisce elevate quantità di dati
- **Online Processing**: Transazioni eseguite in modo interattivo, richiesta e risposta si susseguono immediatamente

Sistemi Client Server

Modello dotato di interfaccia grafica in cui i client mandano richieste di servizio ed i server soddisfano la richiesta ed inviano i dati ai clienti. Elemento centrale del sistema è la rete.

Esistono **3 tipologie di server**:

- Server Dati (DBMS)
- Server Applicazioni
- Server di Sviluppo

Il client per instaurare una connessione necessita di **indirizzo IP** (o lookup DNS) ed un numero di **porta** al quale viene assegnato il servizio.

L'obiettivo del **DNS** è **tradurre** i **nomi** di dominio in **indirizzi IP**, esistono migliaia di server DNS nel mondo strutturati in **una gerarchia a 5 livelli**:

1. Root Level
2. Top Level Domain
3. Second Level Domain
4. Sub-Domain
5. Host

I **Root Level** sono gestiti da **12** diverse organizzazioni contengono **l'elenco globale** dei domini di primo livello (TLD).

I **Top Level Domain** sono costituiti da due categorie:

- **Gerarchia organizzativa** (.com, .net, .org)
- **Gerarchia geografica** (.uk, .fr, .pe)

Sistemi Distribuiti

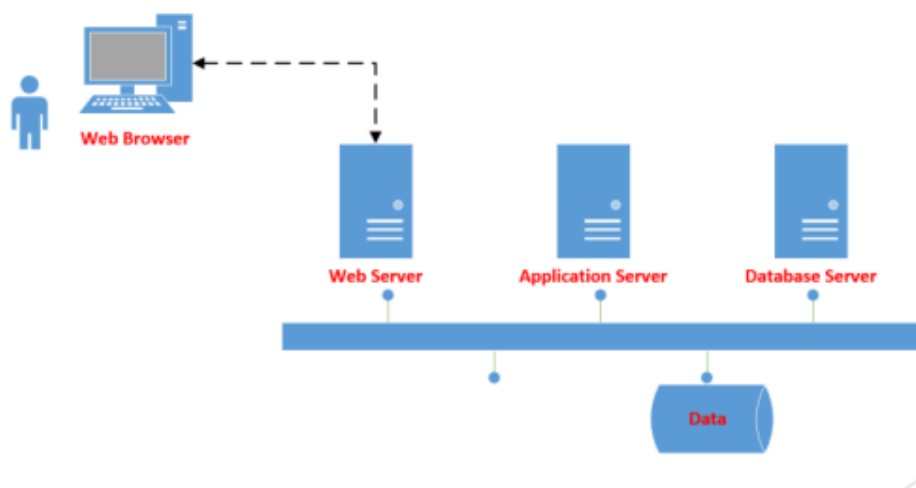
Computer collegati in **rete** (sia LAN che WAN) detti **nodi** che ospitano programmi che condividono **dati** e **risorse**.

Web Application

Passo avanti rispetto al modello Client Server, il client si interfaccia con il server tramite un **browser web**. Alla base del funzionamento di questo modello ci stanno i protocolli **TCP/IP** ed **http**.

Architettura a tre livelli

Evoluzione del modello Client Server, standardizza l'accesso alle applicazioni in maniera sempre più integrata con Internet e gli standard.



Grid Computing

Modello progettato per la **condivisione di risorse su larga scala**, ha alti livelli di **performance** e di **throughput** ed è dotato di una architettura **aperta, modulare ed estensibile** che da una definizione per i servizi di base, una interfaccia alle applicazioni ed i protocolli di interoperabilità delle risorse.

Elaborazione in Cloud

Modello di accesso **via rete** da qualsiasi luogo ad un **insieme di risorse condivise** rilasciate **rapidamente** e con **un costo di gestione minimo**.

Gode delle seguenti **caratteristiche**:

- **On Demand:** Gestione della capacità elaborativa demandata al provider senza necessità di acquisizione e configurazione di nuovo hardware o software.
- **Pooling di risorse condiviso:** Risorse localizzate su piattaforme condivise assegnate in modo dinamico.
- **Elasticità rapida:** Variazione delle risorse immediata per fronteggiare picchi di richieste.
- **Servizio Misurato:** Il provider misura i consumi ed il numero di utenti per fatturare il costo del servizio (a consumo).
- **Accesso via rete:** Da qualsiasi parte del mondo e da qualsiasi device.
- **Alta Disponibilità e Disaster Recovery:** Deve garantire più del 95% di disponibilità e la resistenza ad eventi catastrofici. Questi fattori vengono stabiliti in un contratto chiamato **Service Level Agreement (SLA)**.

Suddiviso in **tre modelli infrastrutturali**:

- **Public Cloud:** Forma canonica, gestione dell'infrastruttura e del software a carico del provider.
- **Private Cloud:** Servizi di calcolo offerti solo ad utenti selezionati e non al pubblico generale. Offrono maggiori livelli di sicurezza.
- **Hybrid Cloud:** Combinazione delle precedenti.

Modelli in Cloud

Esistono 3 tipologie di modelli:

- **Infrastructure as a Service (IaaS):** Fornisce **massima capacità di controllo** da parte dell'utente e consente nel servizio di fornitura di risorse hardware quali server, rete, memorizzazione, archivio e backup.
- **Platform as a Service (PaaS):** Fornisce piattaforme di elaborazione come **middleware** e **web server** ed offre **minori capacità di gestione al cliente**.
- **Software as a Service (SaaS):** Fornisce l'accesso ad **applicazioni web** di tipo custom o package **completamente gestiti dal provider**. **Minime capacità di controllo da parte del cliente** che però può ottenere il servizio tramite **canone di abbonamento**.



Software Custom

Detto anche tailor-made software è **sviluppato appositamente per un unico utilizzatore**, chi lo sviluppa deve tenere in considerazione le fasi di sviluppo ed i costi ad esso associati.

Software Packaged

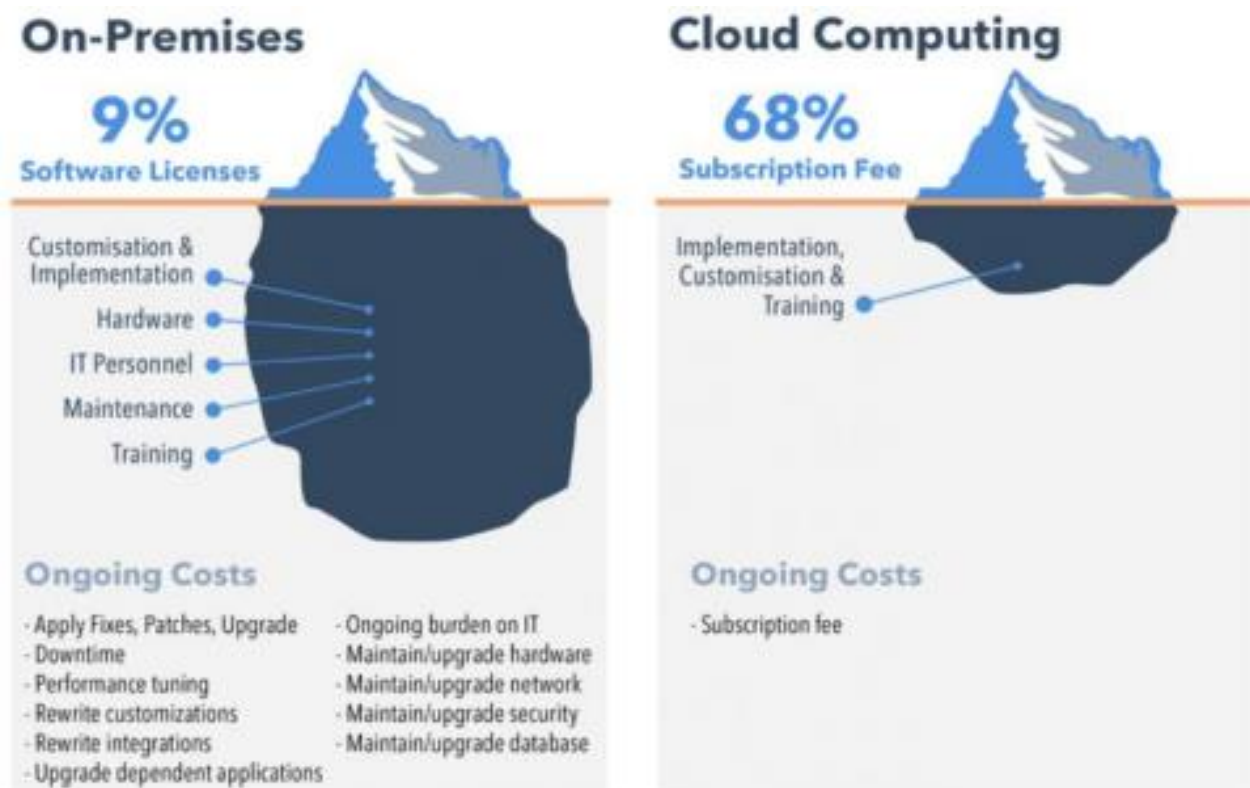
Software sviluppato per essere venduto a molti clienti, funzionalità comuni a tutti i clienti, configurabile tramite parametrizzazione ed aggiunta di moduli.

Software On Premise

Consiste nell'installazione ed esecuzione del software **su architettura locale presso il cliente**. Offre **maggiore controllo esclusivo** su sistemi e dati ed una **gestione interna** dei dati sensibili, **tuttavia**, necessita di uno **staff IT dedicato** ed ha maggiori **costi** di investimento (**CAPEX**) ed un **costo** di **gestione complessivo elevato**.

Software In Cloud

Consiste nell'installazione ed esecuzione del software **su architettura in cloud fornita da un provider**. Raggiungibile tramite rete pubblica, possibile in SaaS installare il software **in proprio** o **acquisirlo dal provider** stesso in **cloud**. Il **controllo** e la **gestione** dei sistemi sono **delegati al provider**, non necessita di staff IT dedicato, ha **costi di gestione inferiori** rispetto al modello on premise in quanto sono considerati costi operativi (**OPEX**).



Alta disponibilità, Continuità Operativa e Scalabilità

Architettura a tre livelli (tier)

Composta da:

- **Livello di presentazione:** Interagisce con client e parte applicativa, mostra le informazioni relative alle richieste utente.
- **Livello applicazione:** Esegue i programmi applicativi che elaborano le richieste utente. Prepara ed inoltra le query al database server.
- **Livello dati:** Disegna la struttura dei dati e ne gestisce l'accesso nascondendo la logica al livello applicazione.

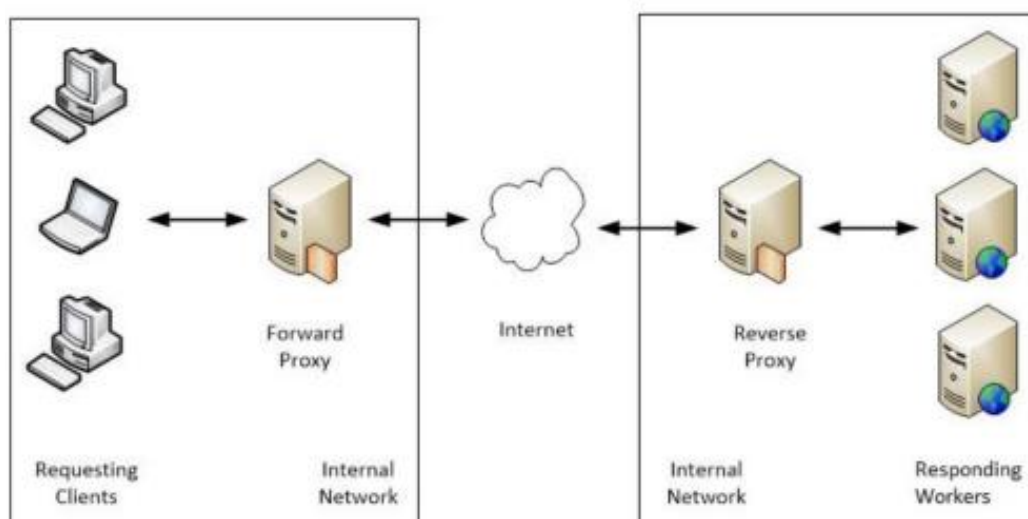
Architettura e sicurezza

Esistono 2 tipologie di proxy:

- **Forward Proxy:** Funge da intermediario tra **client** e la **rete**
- **Reverse Proxy:** Funge da intermediario tra la **rete** ed i **server**

Oltre a proteggere la rete e bilanciare il carico, i proxy fanno caching di tipo:

- **Passivo:** Pagina al quale facciamo l'accesso memorizzata in cache.
- **Attivo:** In periodi di basso carico aggiorna la cache interrogando i server.



Continuità operativa e scalabilità

La **continuità operativa** deve garantire che il sistema rimanga disponibile anche a dronte di **eventi distruttivi** e che eventuali **tempi di indisponibilità** siano **ridotti al minimo** salvaguardando i dati aziendali

La **scalabilità** consiste nella possibilità di adeguare la capacità elaborativa in base al carico, con **scale up** si intende l'aumento della capacità con **scale down** l'inverso.

Alta Disponibilità o High Availability (HA)

Con **alta affidabilità** si intende che i **periodi di fermo imprevisto** del servizio siano **ridotti al minimo**, è un valore solitamente espresso in percentuale che si aggira solitamente al 90% in un mese, un sistema che garantisce **maggiore affidabilità** prevede anche **costi maggiori** per l'utente finale.

Il **70% dei downtime non è pianificato** e dipende principalmente da errori **software, hardware o umani**. Una soluzione di HA deve essere in grado di **fronteggiare** questi **problemi** e **mitigarne gli effetti**.

L'**Alta Affidabilità** si occupa principalmente di **due aspetti**:

- **Disponibilità dei processi**
 - Attivi
 - Accessibili
- **Disponibilità dei dati**
 - Applicativi
 - File di sistema

HA – Disponibilità dei processi

- Servizi di failover => Failover clusters
- Servizi di scalabilità => Load Balancing

HA – Disponibilità dei dati

- Resilient Disk Storage
- Data Replication

Servizi di Failover

Esiste **una sola istanza** di un processo **attiva** ad un certo istante, girano su **cluster** (insieme di nodi o server) **che riavvia i servizi automaticamente** su un nodo attivo in modo **trasparente** e senza **riconfigurazione** client.

Nel caso di **failover** lo switch avviene **automaticamente** ed il downtime è ridotto al minimo, nel caso di **switchover** lo switch è **manuale** e si ha un breve periodo di downtime.

Cluster per HA

Il cluster si compone di uno o più **server fisici** detti **nodi**.

Un **server logico** è un **raggruppamento di risorse** che appare come un server fisico ma che a differenza dei server fisici **può girare su qualsiasi nodo** di un cluster.

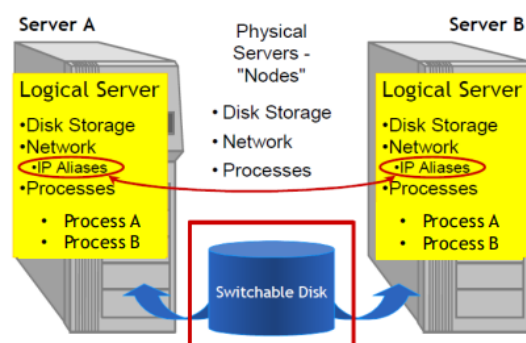
In caso di **malfunzionamento**, il **cluster framework** sposterà il server logico ad un altro nodo attivo, grazie all'utilizzo degli **IP alias** non è necessaria alcuna riconfigurazione rendendo l'operazione **trasparente** agli utenti.

Lo **Switchable Disk** è un requisito fondamentale di un cluster e rappresenta un **volume di dischi condiviso** ed accessibile da tutti i server del cluster, di solito solo un server alla volta **accede al volume condiviso**.

La **Storage Area Network** è una tecnologia che permette di **collegare uno o più dischi** mediante fibra ottica usando il protocollo **SCSI**. Esistono soluzioni in cui l'accesso è consentito a più nodi simultaneamente.

Il **Cluster Management Software** è installato su ogni nodo ed effettua il monitoraggio degli altri nodi del cluster e delle risorse per ciascun nodo. Monitora lo stato degli altri nodi tramite:

- **Heartbeat Connection:** Invii periodici di un messaggio per indicare il normale funzionamento.
- **Ping ICMP:** Invio di ping ai nodi ed attesa di risposta, se non arriva entro un timeout il CMS segnala una anomalia.



Configurazioni Active-Passive (AP) ed Active-Active (AA)

Nella configurazione **Active-Passive** un nodo esegue le applicazioni ed un altro resta in **standby** pronto ad ospitare il **server logico** in caso di **failover**. Sono solitamente utilizzati per sistemi **performance-critical**.

- **Vantaggio:** Nessun rischio di performance
- **Svantaggio:** Maggiori costi, parte dell'hardware inattivo

Nella configurazione **Active-Active** ci sono servizi in esecuzione su **entrambi i nodi simultaneamente**. Sono solitamente utilizzati per sistemi **non-performance-critical**.

- **Vantaggio:** Minori costi, hardware sempre utilizzato.
- **Svantaggio:** Rischio di performance in caso di failover

Sequenza di Failover

La sequenza di failover parte con il tentativo del Cluster Management Software di **riavviare il servizio sullo stesso nodo**, se il nodo non è attivo scatta il **failover** riavviando il **server logico** su un **altro nodo**.

In caso di **failover** si ha una **indisponibilità** breve del servizio di **circa 1 o 5 minuti**, il **riavvio** di tutte le **componenti** e la **riconnessione** dei **client**.

Misurare l'alta affidabilità

Esistono 3 modi per misurare l'alta affidabilità:

- **Expected period of operations:** Stima di operatività annuale.
- **% Availability:** Disponibilità espressa in percentuale.
- **Mean Time Between Failures (MTBF):** Stima predittiva.

Expected period of operations

Rappresenta la **durata totale in ore di lavoro del sistema** ed è la base di calcolo della disponibilità per ottenere il **valore di downtime atteso** in ore.

% Availability

Indicata come la **percentuale di ore** (settimanali, mensili o annuali) nel quale il **sistema deve essere disponibile**.

$$\%availability = \frac{(TempoTotale - SommaTempoNonOperativo)}{TempoTotale}$$

Availabilty	Minimum Expected Time	Maximum Allowable downtime	Remaining time
99.00%	8672	88	0
99.50%	8716	44	0
99.95%	8756	4	0
100.00%	8760	0	0

Mean Time Between Failures

Rappresenta la **stima dei downtime attesi** sulla base delle **rivelazioni precedenti**. Indicatore di tipo **predittivo**.

Quando ci sono **più unità dello stesso tipo** il valore unitario **MTBF** è **diviso** per il **numero di unità** diventando molto più **basso**.

$$MTBF = \frac{SommaTempiAttivitaComponenti}{NumeroGuasti}$$

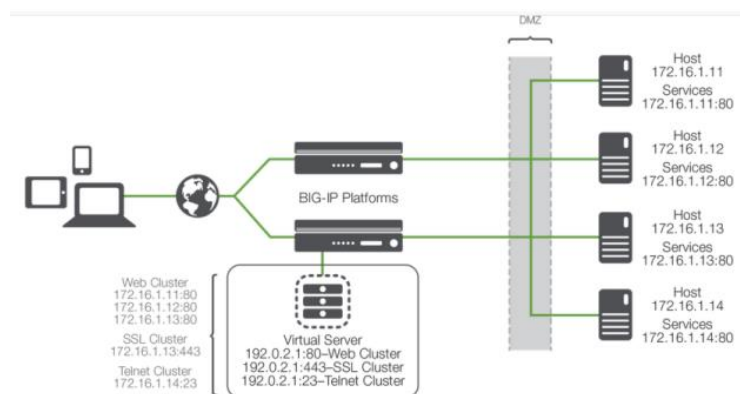
Servizi Scalabili – Load Balancing

Processi applicativi che possono essere **istanziati più volte** nello **stesso server**, in caso di **malfunzionamento** qualunque processo può eseguire le **stesse operazioni**. Le **richieste** vengono **indirizzate al server** secondo politiche di indirizzamento come il **round robin** o il **server meno carico**. Se un server diventa **indisponibile** la connessione sarà **indirizzata ad un altro server attivo**.

Il reindirizzamento di un client allo stesso server viene definito **persistenza della sessione**. In caso di traffico http i bilanciatori di carico ispezionano i **cookie di sessione** per instradare il client allo stesso server.

Il servizio di **Load Balancing** mantiene una **tabella dei server fisici**, **gestisce la politica di carico**, ogni richiesta proveniente dalla rete sarà **distribuita ai server** ed è possibile **definire anche più principi da bilanciare**.

Sequenza SYN – SYN/ACK – ACK



Il client 198.18.0.1 invia una richiesta SYN per una connessione alla porta 80 del server virtuale 192.0.2.1 (il load balancer LB), il suo obiettivo è raggiungere il web server. Il client percepisce esclusivamente il server logico 192.0.2.1:80. Il load balancer, nelle sue tabelle di indirizzamento, ha registrato tre web server che ascoltano sulla porta 80: 172.16.1.11, 172.16.1.12, 172.16.1.13. Dopo aver ricevuto la richiesta, il LB decide di inoltrarla al web server 172.16.1.11:80. Il client 198.18.0.1 continua a vedere solo il server logico 192.0.2.1. In risposta, il web server 172.16.1.11 invia un pacchetto SYN/ACK verso il client 198.18.0.1:5000. Questa comunicazione viene intercettata dal LB 192.0.2.1:80, che a sua volta invia una conferma SYN/ACK al client 198.18.0.1. Infine, il client 198.18.0.1 invia un ACK al LB 192.0.2.1:80, che completa la connessione inoltrando l'ACK al server fisico 172.16.1.11:80.

Politiche di bilanciamento

Esistono 5 tipologie di politiche di bilanciamento:

- **Round Robin:** A rotazione senza ponderare lo stato di carico dei server
- **Weighted Round-Robin:** Server con maggiori capacità di calcolo vedranno assegnati pesi più alti ed avranno una percentuale di carico maggiore.
- **Least-Connections Based Schedules:** Carico distribuito per primo al server con meno connessioni attive.
- **Weighted Least-Connections:** Connessione assegnata alla macchina con il numero minimo di connessioni ed il peso più alto.
- **Locality-Based-Least-Connection:** Politica che tende ad indirizzare allo stesso server le richieste che provengono dallo stesso indirizzo IP

Tipi di load balancer

L'HA deve essere garantita anche per i bilanciatori, ne esistono 2 tipi:

- **Software**
- **Hardware**

Continuità operativa – Graceful Degradation

La **graceful degradation** è un approccio in cui **solo alcune funzionalità sono rese disponibili** al fine di evitare che tutto il sistema diventi inutilizzabile, si privilegiano le applicazioni **critiche** fermando quelle **non critiche**.

Scalabilità orizzontale e verticale

Nella **scalabilità verticale** si aumentano le risorse dei singoli server.

Nella **scalabilità orizzontale** si aumenta il numero di server del cluster.

Continuità dei dati

Esistono 3 diverse tecnologie per la continuità dei dati (**resilient disk storage**) che garantiscono alta affidabilità e continuità dei dati:

- Redundant Array of Inexpensive Devices (**RAID**)
- Storage Area Network (**SAN**)
- **Data Mirroring**

SAN: Implica l'uso dei RAID ed è ideale per **failover** e **cluster** (switch rapido)

Data Mirroring: Consiste in una copia dei dati in una **località remota**. Rappresenta una soluzione di **disaster recovery**.

RAID

Esistono 6 tipologie di RAID:

- **RAID 0:** Striping del disco, non è fault tollerant in quanto non c'è ridondanza ed ha migliori performance.
- **RAID 1:** Mirroring del disco, i dati vengono duplicati garantisce l'alta affidabilità.
- **RAID 3:** Striping + Controllo Errori. Disco riservato solo per parità.
- **RAID 5:** Striping + parità distribuite in tutti i dischi. Dati di parità su disco differente. Buone performance in lettura ed alta disponibilità.
- **RAID 6:** Doppia parità, maggiore protezione di RAID 5 ma unità disco solo per memorizzare la parità.
- **RAID 10 (RAID 1 + 0):** Striping + Mirroring. Alta Affidabilità.

Continuità dei dati - RAID0, RAID1, RAID5, RAID10

Raid 0

	Disk 1	Disk 2	Disk 3	Disk 4	Disk 5
Stripe 1	Block 1	Block 2	Block 3	Block 4	Block 5
Stripe 2	Block 6	Block 7	Block 8	Block 9	Block 10

Raid 1



	Disk 1	Disk 2
Block 1	Block 1	Block 1
Block 2	Block 2	Block 2
Block 3	Block 3	Block 3
Block 4	Block 4	Block 4
Block 5	Block 5	Block 5
Block 6	Block 6	Block 6

Raid 5

	Disk 1	Disk 2	Disk 3	Disk 4	Disk 5
Stripe 1	Block 1	Block 2	Block 3	Block 4	Parity 1-4
Stripe 2	Block 5	Block 6	Block 7	Parity 5-8	Block 8
Stripe 3	Block 9	Block 10	Parity 9-12	Block 11	Block 12

Raid 10

	Array 1		Array 2	
	Disk 1	Disk 2	Disk 3	Disk 4
Stripe 1	Block 1	Block 2	Block 2	Block 1
Stripe 2	Block 3	Block 4	Block 4	Block 3
Stripe 3	Block 5	Block 6	Block 6	Block 5

 UNIVERSITÀ degli STUDI di CATANIA  **Vedi Note**

Storage Area Network

Una tecnologia SAN implementa **storage condiviso tra nodi di un cluster**. Soluzione ideale per **alta disponibilità** e **Load Balancing**. Utilizzano una tecnologia ad alta velocità su fibra ottica (**FC: Fibre Channel**) ed il protocollo **FCP** (Fibre Channel Protocol) o il protocollo **FCOE** (Fibre Channel Over Ethernet) per far **convergere rete e storage** su stessi cablaggi.

Vengono anche utilizzate tecnologie come **iSCSI** per reti aziendali medio-piccole ed **InfiniBand** in ambiente ad elevate prestazioni.

Network Attached Storage

I server **NAS** sono **file server condivisi** in rete plug & play, supportano **Alta Affidabilità** e **Load Balancing** e vengono connessi ai server **tramite LAN**. Sono **poco adatti** ad uso **DBMS** per via delle **performance inferiori**.

Il **File Storage** è un catalogo per i file dati con una struttura gerarchica, spesso utilizzato nei NAS, si accede ai dati indicandone il path su cui è stato archiviato.

Data Mirroring

Componente fondamentale delle configurazioni di **disaster recovery**, consiste nella **copia integrale dei file su host separato**, equivale sostanzialmente alla ridondanza **RAID 1**.

Offre elevate **disponibilità dei dati** a **discapito** di **costi hardware** e di **rete** aggiuntivi nonché un **rallentamento delle operazioni**.

Single Point Of Failure

Elemento la cui perdita risulta nella perdita integrale del servizio.

COMPONENTE	RIMEDI PER ELIMINARE LO SPOF
Server	Failover Cluster, server bilanciati
Load Balancer Hardware	Failover Cluster
Connettività di rete (cablaggio)	Schede di rete LAN ridondanti e sottoreti differenti.
Schede di rete	Schede di rete LAN ridondanti e sottoreti differenti.
Disco di sistema (OS root disk)	Dischi RAID
Disco dati	Dischi RAID
Alimentazione elettrica (power source)	Alimentazione elettrica sdoppiata e gruppi di continuità (UPS) su entrambe
Scheda di interfacciamento al disco	Schede ridondanti secondo il modello di disco in uso
Sistema Operativo	Failover e sistema applicativo disegnato per il riavvio automatico
Software applicativo	Failover e sistema applicativo disegnato per il riavvio automatico
Errore umano	1. Failover 2. Automatizzare tutto il possibile. 3. Documentare le procedure di emergenza

Backup e Recovery

Il Database Backup consiste nel salvataggio dei file fisici per memorizzare e ripristinare il database su disco o altro supporto offline. Questi file sono:

- **Data File:** Dati e metadati del DBMS
- **Control File:** Definisce caratteristiche dell'istanza del DB
- **Redo/Journal Logs:** Contiene le modifiche effettuate al DB

Esistono diversi tipi di dispositivi e supporti adatti al backup selezionati in base a **criteri** quali: **Capacità**, **Velocità** e **Prezzo**. Alcuni esempi sono: Lettori di supporti magnetici, **Hard Disk**, SAN, CD-ROM, DVD, **SSD**.

Schemi di Backup

Esistono 3 schemi di backup principali:

- **Full Backup:** Copia integrale dei dati. Sempre lento.
- **Incremental Backup:** Salva solo file nuovi o modificati rispetto all'ultimo backup. Backup rapido, restore lento.
- **Differential Backup:** Salva solo file nuovi o modificati rispetto all'ultimo full backup. Restore rapido, backup lento e richiede più spazio.

Full Plus Incremental Backup Schema							
	Day1	Day2	Day3	Day4	Day5	Day6	Day7
File 1	x	x					
File 2	x		x				
File 3	x			x			
File 4	x				x		

Full Plus Differential Backup Schema							
	Day1	Day2	Day3	Day4	Day5	Day6	Day7
File 1	x	x	x	x	x		
File 2	x		x	x	x		
File 3	x			x	x		
File 4	x				x		

Recovery

Attività critica, non funziona quasi **mai**, tuttavia i backup sono inutili senza un efficace recovery. Necessario effettuare **test periodici** e **documentare** accuratamente tutte **le procedure**, conservando una **copia di backup** in un sito remoto e sicuro.

Disaster Recovery

Un **disastro** è un evento che causa il **blocco dell'operatività** dei sistemi informatici procurando un danno esteso e permanente alle strutture di calcolo.

Causato da: Terremoti, Inondazioni, Incendi, Blackout, Attacchi terroristici...

Disaster Recovery – Classificazione dei sistemi

Esistono 4 tipologie di sistemi:

- **Critical**
- **Vital**
- **Sensitive**
- **Non Sensitive**

Nei sistemi **critical** il sistema va rimpiazzato con uno identico, le operazioni non sono riproducibili manualmente e la tolleranza all'indisponibilità è molto bassa. Costo di inoperatività molto elevato.

Nei sistemi **vital** le funzioni possono essere eseguite manualmente per tempi molto brevi, hanno una bassa tolleranza all'indisponibilità ed hanno un costo di inoperatività alto con attesa di ripristino fino a 5 giorni.

Nei sistemi **sensitive** le funzioni possono essere eseguite manualmente per periodi di tempo prolungati, hanno una media tolleranza alla indisponibilità ed un costo di inoperatività medio.

Nei sistemi **non sensitive** le funzioni possono essere eseguite manualmente con basso impatto ed il riallineamento richiede pochi sforzi.

Business Continuity Plan (Piano Dettagliato di Continuità)

Si definisce sulla base di una strategia complessiva di continuità dei servizi deve contenere:

- **Risk Assessment:** Identifica i sistemi che supportano i processi critici
- **Business Impact:** Analizza il costo della perdita dei sistemi critici

Bisogna inoltre sviluppare il **piano per il ripristino delle funzionalità IT** (Detailed Recovery Plan o **DRP**) ed infine un **piano per far funzionare i processi critici** in un periodo di crisi (Business Continuity Plan o **BCP**)

La regola del backup 3-2-1

La regola del backup 3-2-1 stabilisce che:

- Ci siano sempre **3 copie dei dati**
- Si utilizzino **2 diverse tecnologie di memorizzazione**
- **Una copia dei dati offline**

Dimensionamento e Topologia

Dimensionamento del sistema IT

La stima del dimensionamento dell'hardware detto anche **sizing exercise** risponde a 3 esigenze principali:

- Definire una stima per il **budget** di processo
- Effettuare l'**acquisto dell'infrastruttura hardware**
- Disegnare la **topologia complessiva del sistema**

Sizing Exercise

Il sizing exercise viene effettuato in 2 occasioni:

- **Prima dell'inizio del progetto:** Fornisce elementi per l'approvazione ed il bilancio tra costi e benefici
- **Ad inizio progetto:** Per l'acquisto di hardware ed il disegno della topologia

Il **sizing iniziale** permette di stimare in modo grossolano il budget per la componente hardware di un progetto informatico.

Si raccolgono dati come:

- Numero di utenti,
- Numero di accessi concorrenti,
- Complessità della transazione online,
- Tipi di dati principali (clienti, ordini, fatture),
- Numerosità di ogni tipo di dato,
- Tipo, Dimensione e numero di file allegati,
- Richieste di alta disponibilità e disaster recovery.

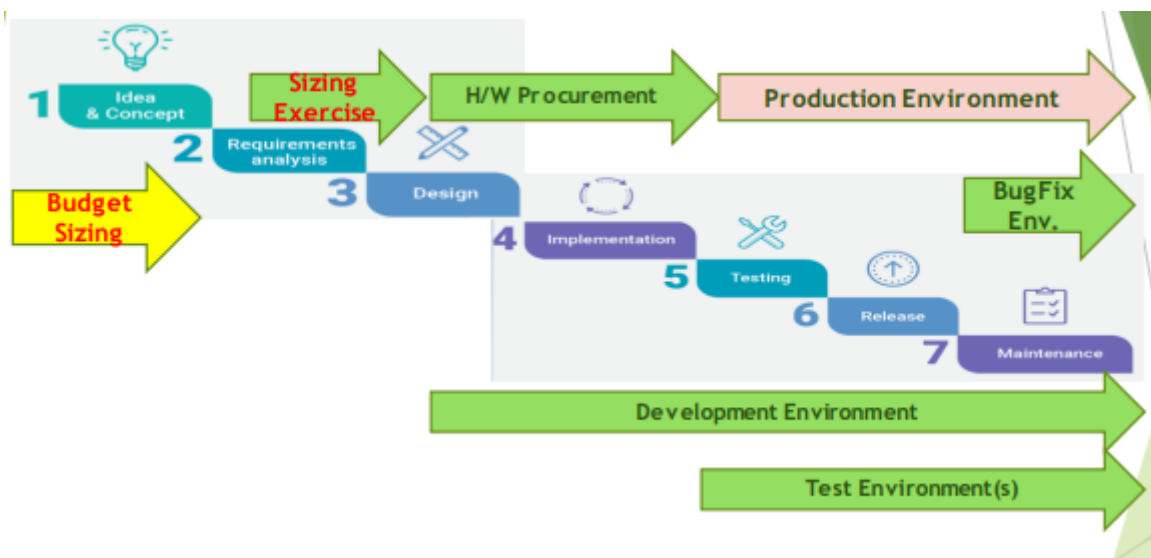
Acquisto dell'infrastruttura

Il **sizing ad inizio progetto** è un passo fondamentale per l'acquisto dell'hardware necessario entro i tempi previsti, il **processo di acquisto** deve essere **compatibile con i tempi di progetto**. Il sizing deve iniziare non appena i **requisiti** sono **sufficienti a stimare il carico**, i requisiti dell'applicazione e la **natura dei dati**. Otteniamo così la **dimensione** e la **potenza dell'hardware richiesto**.

Ciclo di sviluppo

Nel ciclo di vita di un sistema informatico si identificano i seguenti ambienti:

- **Development Enviroment (DEV):** Ambiente di dimensioni ridotte, si usa per unit test che non richiedono un carico elevato. Rimane sempre attivo, anche dopo la fase di rilascio viene utilizzato per l'applicazione e la correzione dei bug.
- **Test Enviroments**
 - **System Integration Test:** Consente il test end-to-end.
 - **User Acceptance Test Enviroment:** Ambiente dove gli utenti finali effettuano il test di accettazione dell'applicazione.
 - **Performance/Stress Test Enviroment:** Ambiente dove si effettuano i test finali di carico del sistema (**stress test**), nella pratica per limitare i costi si usa un ambiente dedicato più piccolo sul quale poi si effettuano proiezioni di carico sul sistema di produzione.
- **Training Enviroment:** Ambiente completo in scala più piccolo dove vengono addestrati gli utenti finale.
- **Production Enviroment:** Deve essere pronto con anticipo sufficiente alla data di inizio della produzione per consentire il caricamento iniziale dei dati
- **Bug Fixing Enviroment:** Ambiente dedicato rispetto a DEV di dimensioni ridotte perché in caso di errore grave l'ambiente di sviluppo potrebbe essere già ad una versione successiva degli sviluppi rendendo impossibile la correzione del bug.



Versioni del software

XX.YY.ZZ.tt[nnnn]

- **X: Major release** se cambia c'è l'evoluzione dell'**architettura** complessiva **dell'applicazione**.
- **Y: Versione dell'applicazione**, varia quando vengono aggiunte **nuove funzionalità**.
- **Z: Patch set**, blocchi periodici di **correzioni funzionalità esistenti**. Non vengono aggiunte nuove funzionalità.
- **Tt: Quick fix**, Modifica urgente per **correggere subito bug critici**.
- **[nnnn]: Build**, non viene apportata alcuna modifica al software ma solo a **librerie di terze parti**.

Requisiti di scalabilità del software

Esistono 2 casi nella pratica:

- **Benchmark** di scalabilità **esistenti** del fornitore del package software
- Software per il quale **non esistono benchmark di scalabilità**

I **benchmark** mostrano il **valore atteso di comportamento** in base a:

- Numero di utenti o transazioni per processore,
- Mbyte di RAM per utente sulla piattaforma
- Numero di core e frequenza della RAM
- Descrizione degli script di test

Quando non si hanno benchmark occorre **effettuare test di carico** con dei prototipi facendo riferimento a **tabelle di performance di riferimento** pubblicate dalla Standard Performance Evaluation Corporation (SPEC).

Stima delle CPU e della memoria RAM

Processo **euristico** che richiede di effettuare **assunzioni** basate sul **numero di utenti** e sulla **percentuale di utilizzo del processore** nel benchmark, se possibile è sempre meglio valutare il **caso peggiorativo** della condizione di esecuzione.

Sempre consigliato aggiungere una maggiorazione o **contingency** del 25% alla stima del dimensionamento ottenuto con l'esercizio di sizing.

Dimensionamento del Database

Per il dimensionamento del database bisogna considerare **CPU** e memoria **RAM**, il DBMS richiede spesso l'utilizzo della **memoria centrale** rallentando l'esecuzione delle operazioni. Occorre considerare i seguenti elementi:

- Numero di connessioni utente
- Numero di connessioni server
- Quantità di memoria unitaria per connessione
- Memoria per i processi interni del DBMS
- Memoria per i processi di sistema Operativo
- Contingency

Dimensionamento del file system

Area del disco dove vengono **memorizzati file rilevanti** per l'applicazione. Occorre considerare i seguenti elementi:

- Tipi di file che devono essere conservati
- Numero di file per ogni tipo
- Dimensione di ogni file
- Fattore crescita ad uno o due anni
- Numero di anni da tenere in linea i dati prima della cancellazione

Crescita dei volumi e capacity plan

Nel **corso della vita** di un sistema è necessario **stimare il sizing** in caso di:

- Rilascio di **nuove funzionalità**
- **Aggiunta del numero di utenti**
- Sistema in **sofferenza di prestazioni**
- Combinazione dei precedenti

Tale esercizio viene definito: **Capacity Plan**.

Per effettuarlo, si **osserva il carico attuale** del sistema, si stabilisce la **capacità residua (headroom)** per ospitare ulteriori carichi e la **capacità addizionale richiesta** rispetto al carico attuale.

Il **nuovo hardware da acquistare** va **stimato** sulla base **dell'headroom** residuo e la **nuova capacità totale richiesta**.

Health Check

La fase iniziale di verifica dello stato di carico prende il nome di **health check** e si effettua monitorando per **lunghi periodi di tempo le risorse** ottenendo un report riepilogativo per ogni server del sistema.

Può essere effettuato:

- Tramite **tool di monitoraggio**
- Tramite **script ad hoc**
- Con **frequenza prefissata**: trimestrale, semestrale, annuale
- **On Demand** nel caso di peggioramenti di performance

Riepilogo del Capacity Plan

Il **capacity plan** si effettua ogni qualvolta sia necessario **ridimensionare il sistema**, si osserva dapprima **il carico attuale** stabilendo la capacità residua (**Health Check**). Si **stima** la **capacità addizionale** richiesta e sulla base di questi 2 fattori è possibile **stimare il nuovo hardware necessario**. Meglio tenere sempre un margine di riserva nelle capacità delle risorse (**headroom**).

Stress Test

Lo stress test è in grado di **individuare**:

- **Limitazioni del dimensionamento** del sistema
- **Necessità di miglioramento (tuning)** dell'applicazione

Contiene almeno i seguenti **passi principali**:

- **Pianificazione dello stress test**
- Creazione di uno **script di automazione**: Simulano **attività utente o carichi entranti**
- Generazione di **dati di test** per scenari definiti
- Esecuzione dei **run multipli**
- Esaminazione dei risultati e identificazione di **colli di bottiglia** (bottleneck)

Quando l'applicazione è ben regolata (**tuned**) si ripetono i cicli di test per identificare se esistono **limiti nell'hardware**.

Stress Test: Esecuzione e Tuning

L'esecuzione dello stress test porta ad ottenere anche il tuning dell'applicazione. Una performance inferiore è dovuta alla necessità di:

- Tuning dell'applicazione
- Tuning del database
- Adeguamento Software

Ramp Up e Think Time

Il **ramp up** rappresenta il fronte di salita iniziale di richieste verso il sistema.

Il **think time** è il tempo medio che intercorre tra 2 richieste utente. Indica la **dinamicità** degli utenti.

Stress Test: Fasi tipiche

- **Preparazione dello script di test**
- Esecuzione **stress test** fino al raggiungimento del limite delle capacità
- Esecuzione **tuning applicativo**
- Esecuzione **tuning database**
- Verifica/rettifica parametri **think time e ramp-up**
- Esecuzione stress test fino al raggiungimento del limite delle capacità

Stress Test: considerazioni

La **potenza dell'ambiente di stress test** deve essere **paragonabile** a quella di **Produzione**, in alternativa è possibile **proiettare la performance** del sistema **ridotto** su quello di **produzione**. Effettuare test accurati prima di entrare in servizio è **cruciale** per la riuscita del progetto **riducendo** il **rischio di gravi malfunzionamenti**.

Definizione di Topologia

La topologia del sistema è il **disegno architettuale** che gestisce la potenza di calcolo tra più **macchine fisiche** in **load balancing**, **cluster** o dedicate a specifici sottosistemi. Dopo aver ultimato l'esercizio di sizing occorre pertanto **stimare tutti i requisiti specifici** per definire la **topologia del sistema**.

Fattori che influenzano la topologia

I principali fattori che influenzano la topologia del sistema sono:

- **Disponibilità del sistema:** Tolleranza a guasti e malfunzionamenti mantenendo la disponibilità del sistema.
- **Scalabilità:** Il sistema deve rispondere alle necessità di calcolo adeguandosi rapidamente sia **orizzontalmente** che **verticalmente** alle esigenze
- **Fault Tolerance:** Guasti ai server non devono interrompere il servizio ma farlo scendere ad un livello di prestazione accettabile (**graceful degradation**)
- **Segregazione delle funzionalità e degli utenti:** Utenti differenti devono avere risorse hardware differenti.

Virtualizzazione

Il concetto della virtualizzazione

La **virtualizzazione** consiste **nell'astrazione dei componenti fisici** in un **oggetto logico** con il beneficio di allocare in modo più **efficiente** ed **agevole** una risorsa fisica (**consolidamento dei server**) che viene **suddivisa** in **risorse logiche** che l'utilizzatore vede come una **risorsa fisica**.

Una macchina virtuale può virtualizzare **tutti i tipi di risorse hardware**, il **software** che fornisce l'ambiente in cui operano le **macchine virtuali** è detto **Hypervisor**.

Un **Hypervisor** deve **soddisfare 3 condizioni**:

- **Fedeltà**: Ambiente virtualizzato identico all'originale
- **Sicurezza**: Controllo completo delle risorse di sistema
- **Prestazioni**: Differenza di prestazioni tra ambiente virtualizzato ed originale ridotta al minimo.

Esistono **2 tipi di Hypervisor** o VMM:

- **Tipo 1**: Installato direttamente sull'hardware o bare-metal
- **Tipo 2**: Software che vive tra sistema operativo e macchine virtuali

L'importanza della virtualizzazione

I **sistemi virtuali** sono **set di files** che possono essere **copiati** e **spostati** che garantiscono maggiore **flessibilità di calcolo** ed una **elevata disponibilità** durante i downtime.

Il **consolidamento dei server** implica un **maggiore risparmio energetico** in quanto si riduce il numero di server fisici ed un **maggiore risparmio dei costi del personale**.

Inoltre, la virtualizzazione garantisce **disaster recovery** e richiede lo spostamento dei carichi di lavoro in un altro sito.

Le **macchine** possono essere **spostate** da un **host fisico ad un altro senza interruzioni** ed è inoltre possibile **aggiungere** e **rimuovere risorse** “a caldo” senza influire sul tempo di **disponibilità**. In caso di **disastro** grazie alla copia delle macchine virtuali l'intero sistema può essere **ripristinato in poche ore o minuti**.

I trend ed il cloud computing

La **virtualizzazione** è il motore del **cloud computing**, con la virtualizzazione le **risorse** possono essere **gestite quasi automaticamente** in modo altamente **scalabile** e **disponibile**. **Riducono i costi amministrativi** e rendono **rapida** la **fornitura** di ambienti di calcolo (**provisioning**). Semplificano inoltre la delivery di nuove applicazioni rendendo più **rapida** la loro **implementazione** senza sacrificare **scalabilità, resilienza o disponibilità**.

Hypervisor

Software che funge da **arbitro di risorse** che **astrae il livello fisico**, si trova tra le risorse fisiche e le macchine virtuali di un server, forniscono un **ambiente identico all'ambiente fisico** consumando in modo **limitato** le risorse del sistema e **mantenendone il controllo completo**.

Hypervisor Type 1

Hypervisor che viene **eseguito direttamente sull'hardware** senza un sistema operativo sottostante, viene definita anche implementazione **bare-metal**, a differenza del tipo 2 garantisce **massima efficienza**.

Hypervisor Type 2

Hypervisor **meno efficiente** perché è **installato** ed opera **sul sistema operativo** del server fisico, a differenza del tipo 1 quest'ultimo deve **consegnare la richiesta al sistema operativo** che gestisce le richieste di I/O. Il vantaggio è che è possibile **supportare più tipi di hardware** purché mediati dal Sistema Operativo, Di contro il Sistema Operativo rappresenta un **Single Point Of Failure (SPOF)**.

KVM e QEMU

KVM è un hypervisor di **tipo 1** mentre **QEMU** è un hypervisor di **tipo 2**.

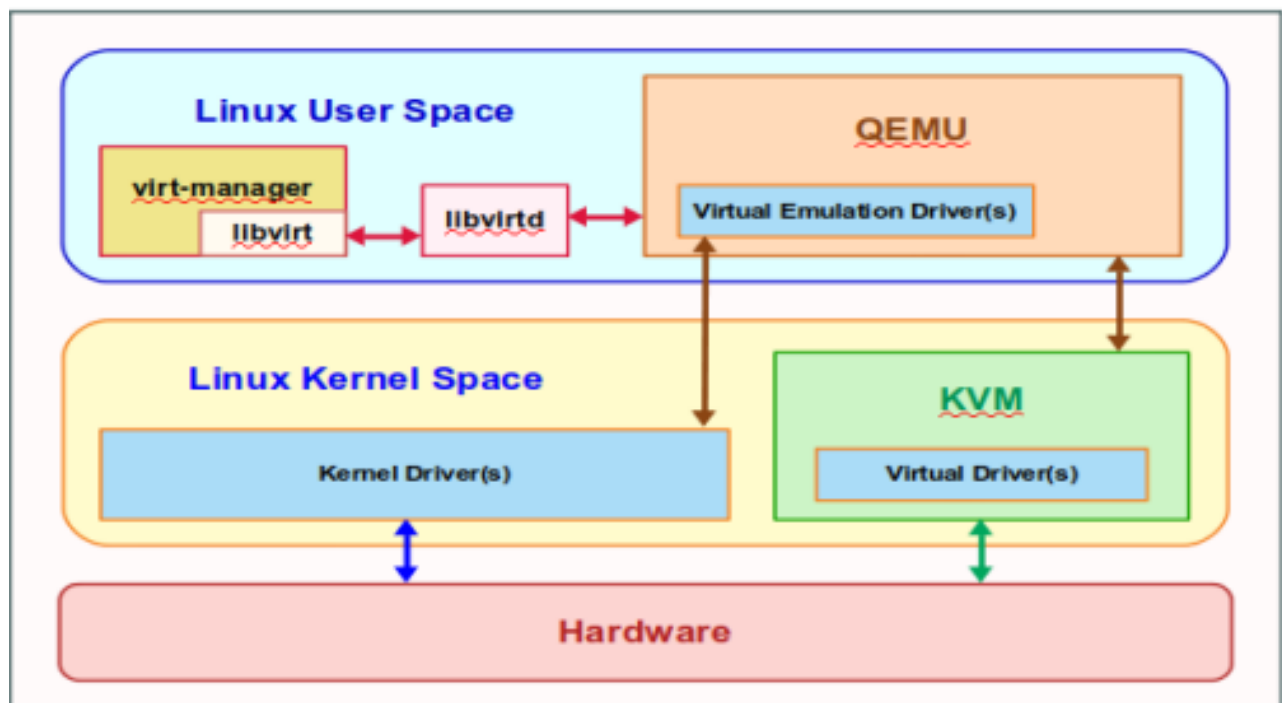
KVM è un modulo del kernel Linux compatibile su piattaforme x86.

QEMU offre al guest una piattaforma differente da quella host ed utilizza la tecnica del **Dynamic Binary Translation**, cioè, che le istruzioni eseguite dal guest vengono opportunamente **tradotte** dall'hypervisor. Processo di traduzione **inefficiente** rispetto alla virtualizzazione nativa.

QEMU si rende necessario per tutto ciò che concerne la **gestione delle VM** e sfruttando le capabilities del modulo KVM insieme a QEMU si ottiene la **virtualizzazione di tipo 1** mediante estensioni per la virtualizzazione della CPU.

Libvirt è una libreria che permette di **scrivere tool** con interfacciamento diretto a **tecnologie di virtualizzazione** quali KVM e QEMU.

Virt-manager è una interfaccia grafica per la gestione delle macchine virtuali che usa **libvirt**.



Supporto hardware per la virtualizzazione

I sistemi operativi garantiscono l'accesso alle risorse mediante gli anelli di protezione o protection rings. I livelli di protezione sono codificati con 2 bit:

- **Level 0:** Sistema Operativo
- **Level 1:** Poco usato
- **Level 2:** Poco usato
- **Level 3:** Applicazioni

Le istruzioni **di livello 0** sono assegnate **al Sistema Operativo host**; tuttavia, il **Sistema Operativo guest** si aspetta gli **stessi privilegi** che in realtà **non possiede** e presuppone il **controllo totale** del suo **spazio di indirizzamento** in memoria.

Problemi:

- **Ring Aliasing:** Il SO guest vuole **eseguire operazioni senza possedere i necessari privilegi**
- Spazio di **indirizzamento** del SO guest **limitato**
- Non tutte le istruzioni richiedono un **livello 0 per l'esecuzione**
- Istruzioni della CPU concepite per passaggio veloce da user-space a kernel-space **compromettono l'efficienza** a causa del VMM

Soluzione iniziale fu la **paravirtualizzazione** che consentiva di modificare il kernel del sistema operativo guest ma aveva **forti limitazioni di compatibilità con i guest OS**.

Intel sviluppò nel 2006 una soluzione per dare supporto diretto alla virtualizzazione come l'architettura VT-X che introduce nuove modalità operative per il codice eseguito dalla CPU:

- **VMX root:** Modalità operativa hypervisor
- **VMX non root:** modalità operativa guest OS

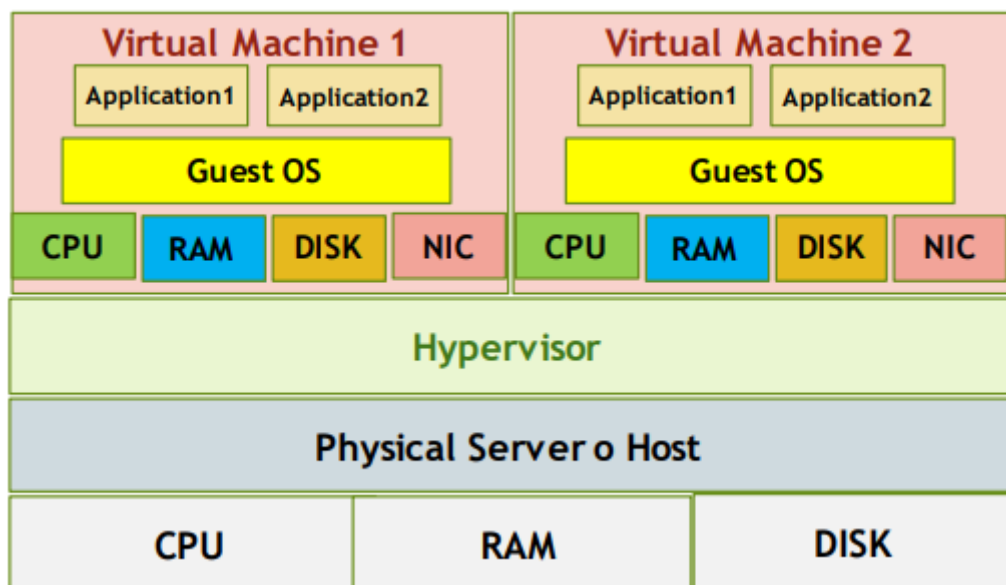
Due transizioni:

- **VM entry:** VMX root -> VMX non root
- **VM exit:** VMX non root -> VMX root

La **Virtual Machine Control Structure** contiene informazioni sulle **transazioni di stato** (VM entry, VM exit), rappresenta una **soluzione ai problemi visti in precedenza**.

Hypervisor – Allocazione delle risorse

Un hypervisor si comporta come un **sistema operativo** in grado di gestire **interi server virtuali**, gestisce tutte le **richieste di I/O** di archiviazione e l'**I/O di rete**, **l'elaborazione della memoria** e la **CPU**. Alcuni hypervisor hanno opzioni per **dare priorità** ai diversi **guest**. Questa idea di gestione e allocazione delle risorse è fondamentale per determinare la configurazione dell'hardware fisico. Sempre meglio avere degli **extra disponibili** per gestire **picchi di performance**.



Virtual Machine

Componenti fondamentali della **virtualizzazione**, sono i contenitori dei sistemi operativi ed i programmi applicativi, sono eseguiti da un hypervisor che funge da **interfaccia** tra **hardware** e **dispositivi virtuali**.

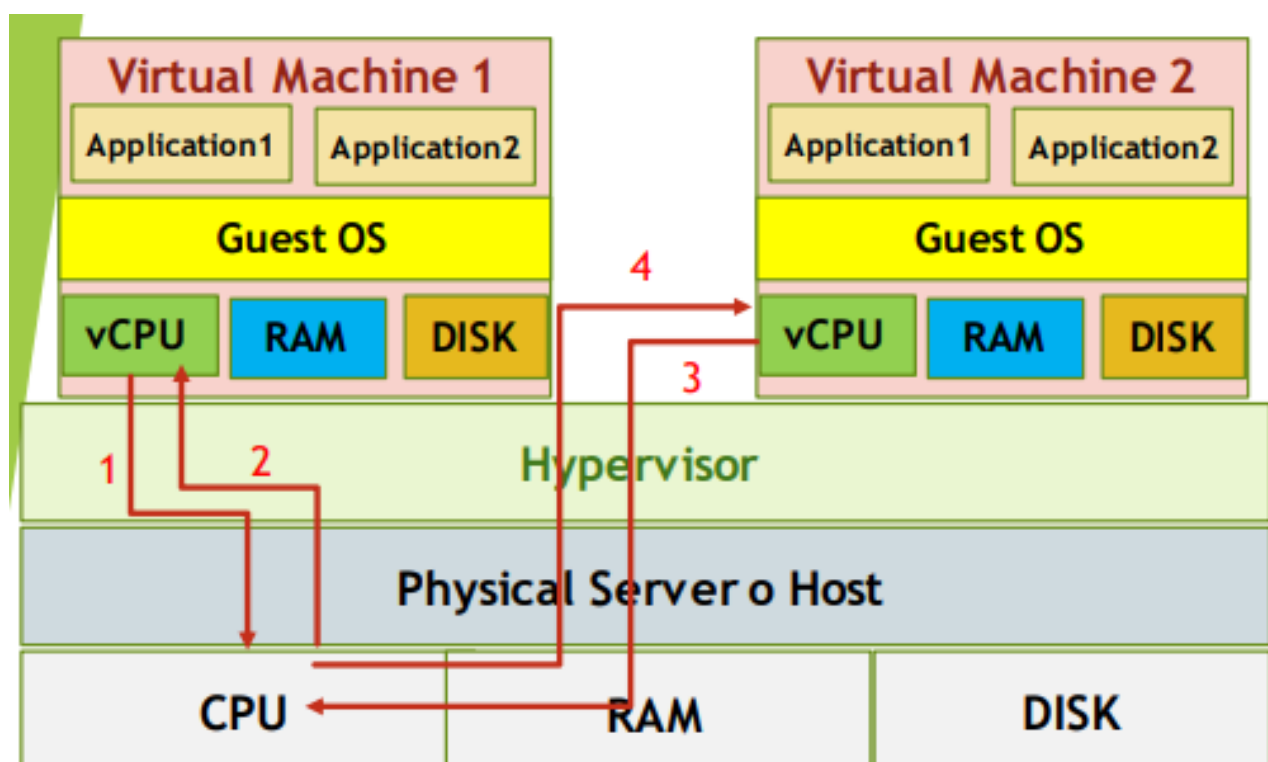
Una macchina virtuale si comporta come **server fisico** e appare alle applicazioni come tale. Il sistema operativo, le app, le connessioni di rete e quant'altro è raggruppato in un **set di file**. Questo **packaging** rende le macchine virtuali **flessibili** e **gestibili** attraverso l'utilizzo dei file. Le macchine virtuali possono essere **clonate**, **aggiornate** e **spostate** da un luogo all'altro **senza** mai dover **interrompere** le applicazioni dell'utente.

Gestione della CPU della VM

La **virtualizzazione del processore** è fondamentale per **ottenere buone prestazioni** della macchina virtuale, la **gestione** è **cruciale** per garantire **performance** paragonabili a quella delle **CPU fisiche** ed un uso efficace delle **risorse disponibili**.

L'hypervisor assegna degli **intervalli di tempo sui processori disponibili** nel server host fisico che prendono il nome di **vCPU**, ogni volta che **l'hypervisor** schedula una **sessione per la vCPU**, la **CPU fisica** esegue le istruzioni e le **restituisce all'hypervisor** che li restituisce alla **macchina virtuale**. Le CPU virtuali **non sono associate** a CPU fisiche l'hypervisor sceglierà **qualsiasi CPU fisica disponibile**. Ogni guest utilizza una parte di una CPU fisica e quindi si può **allocare più di una vCPU** per ogni **CPU fisica** del server.

Gli hypervisor consentono **l'aggiunta** (e la rimozione) di **vCPU 'a caldo'** senza spegnere la macchina virtuale, i sistemi operativi supportano **l'aggiunta di CPU a caldo** ma **non** è ancora possibile **rimuovere CPU a caldo** per tutti i Sistemi Operativi.



Gestione della RAM della VM

Risorsa cruciale e **soggetta ad elevata richiesta**, gli hypervisor la gestiscono estraendo **blocchi di dati** tra memoria fisica del server e ciò che è stato allocato alle macchine virtuali. Viene decisa quanta **RAM** allocare **in fase di definizione** della macchina virtuale che vede il quantitativo di memoria allocato alla creazione **indipendentemente dalla RAM fisica installata nell'host**. Anche se a una macchina virtuale viene allocata una quantità di memoria, questa non è **riservata realmente per la VM**.

Una tecnica utilizzata per **recuperare la memoria** dalla macchina virtuale si chiama **ballooning**:

- Per recuperare la memoria fisica da una macchina virtuale, le **pagine** in memoria devono essere **ripristinate su un altro dispositivo** di archiviazione, in questo caso l'area di paging del disco.
- Il **driver baloon** viene attivato e (praticamente) **gonfiato**, costringendo il sistema operativo a **scaricare le pagine dalla memoria**.
- Il **sistema operativo** sceglie le **pagine da svuotare** perché sa quali sono le pagine **utilizzate meno di recente** e sono obsolete, rendendole buone candidate da rimuovere.
- Una volta **svuotate le pagine**, il driver del baloon si **sgonfia** e l'hypervisor **recupera la memoria fisica** per l'uso. Di solito, questo processo si verifica solo quando c'è contesa per la memoria riservata fissa alla VM.

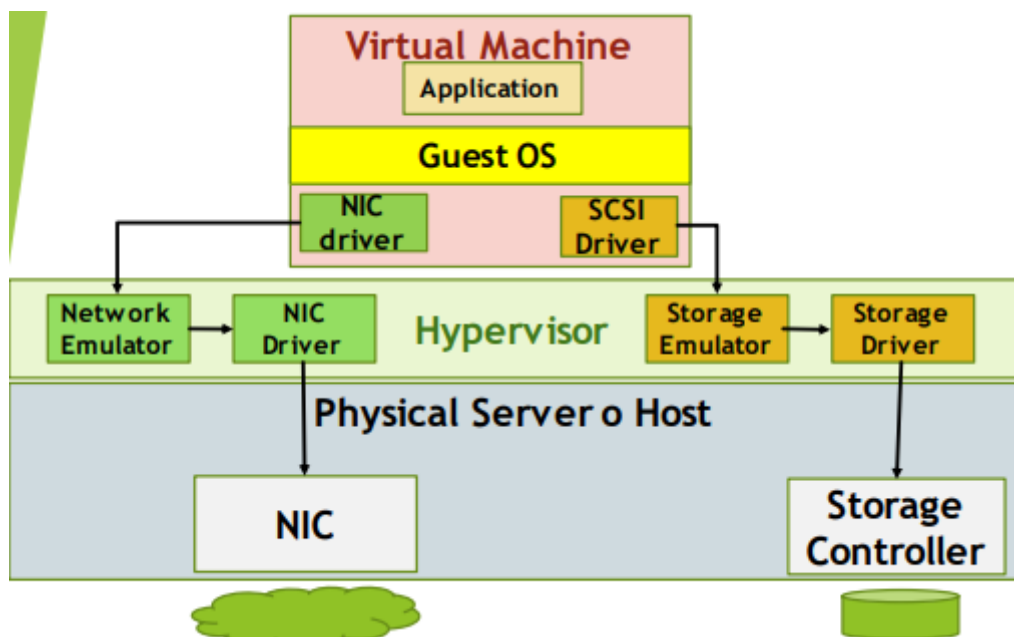
Memory Overhead dell'hypervisor

L'hypervisor impone un **overhead** (consumo addizionale di risorse) al **consumo di RAM**. Per ogni macchina virtuale viene riservata una **piccola porzione di memoria addizionale** e della memoria allocata per l'utilizzo della macchina virtuale **utilizzata per funzioni operative come le tabelle di mappatura della memoria**.

Gestione dello storage della VM

Astrazione delle risorse fisiche **visibile alla VM** come **unità fisica** allocata durante la creazione della macchina virtuale, possibile estenderla dopo l'allocazione e risiede in uno più file fisici dell'host.

La macchina virtuale si interfaccia con un **emulatore SCSI** che riceve la richiesta di I/O ai dischi e la mette in una **coda di richieste** che vengono successivamente passate al driver che è collegato al controller dell'host fisico. Il controller esegue la richiesta e riceve i blocchi di dati. I blocchi di dati seguono quindi il **percorso inverso**.



Virtual Networking della VM

Una rete virtuale è una **rete privata** definita dal software in un ambiente virtuale, gli **switch** di reti virtuali sono utilizzati per la **costituzione di sottoreti virtuali**. Strategia comune per **proteggere** applicazioni e server virtuali da **attacchi indesiderati**. Possibile anche aggiungere **switch virtuali** per separare le reti migliorando le **prestazioni** e riducendo il traffico su ogni segmento.

Nel caso in cui il traffico dati dello storage viaggi su una rete è necessario allocare una **larghezza di banda adeguata** per evitare problemi di **prestazioni**. Gli switch virtuali forniscono **segmentazione** ed **isolamento** del traffico di rete garantendo **sicurezza** ed **integrità** dei dati.

Clone e copia di una VM

La **Macchina Virtuale** è un **insieme di file** che contengono:

- Configurazione della macchina virtuale
- Sistema Operativo
- Programmi Applicativi
- Dati Utente

Fare un backup di una macchina virtuale equivale a **clonarla**, è possibile istanziare una macchina da un **clone** o da un **template** ed è anche possibile salvare la macchina in uno **stato particolare** creando uno **snapshot**.

La virtualizzazione consente operazioni di **manutenzione**, **backup** e **recovery** più **tempestive** e **meno complicate** rispetto alle stesse attività svolte su macchine fisiche, la distribuzione di una macchina virtuale può richiedere solo pochi minuti.

Gli **snapshot** permettono di **ripristinare** una macchina allo stesso punto di partenza consentendo di testare gli aggiornamenti del software in un ambiente esistente ed eventualmente tornare indietro ad un punto specifico nel tempo (**rollback**).

Il formato standard per i file di macchine virtuali è **l'OVF** che definisce un formato standard per **packaging** e **distribuzione**.

Una macchina virtuale può essere creata a partire da un **template** che include **sistema operativo**, **pacchetti software** e **configurazione hardware**. A differenza della **clonazione** dove l'origine è una macchina virtuale qui si parla di **modelli** ovvero **macchine virtuali non avviabili**.

Snapshot di una VM

Gli **snapshot** consistono nel **salvataggio di una macchina virtuale** in un determinato momento nel **tempo**, sono particolarmente utili in ambienti di **test** e **sviluppo** ed è possibile usarle per **ripristinare i server virtuali**.

Si differenzia dai **template** perché oltre alla semplice definizione della macchina virtuale contiene anche i **dati** e lo **stato della macchina virtuale**, informazioni che in un **template non esistono**. **Non sono utilizzabili** ai fini di **backup** perché provocano un **degrado delle prestazioni** della macchina guest.

Disponibilità del sistema virtualizzato

Le architetture virtuali utilizzano una serie di strategie per evitare guasti hardware che causino la caduta di una macchina virtuale.

Esistono tre livelli da considerare

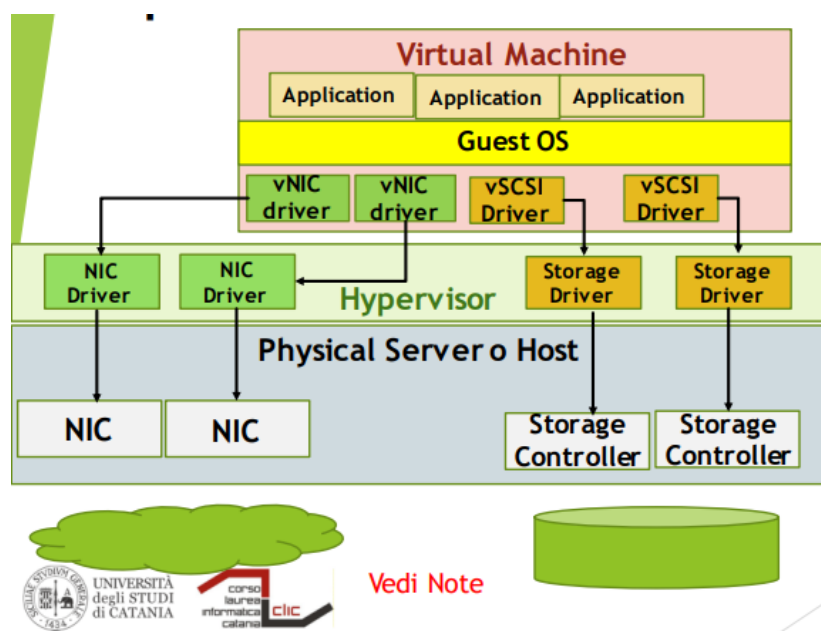
- Una singola macchina virtuale
- Un host o gruppi di host
- Intero data center

RAID + Multipathing

Il **percorso fisico viene duplicato** tramite due controller nell'array di disco ed analogamente si possono **duplicare le schede di rete virtuali**, se un componente fallisce resta **ancora un percorso a disposizione**. Il **multipathing** è **trasparente** per il sistema operativo e per tutte le applicazioni ed offre **vantaggi in termini di prestazioni** bilanciando il carico tra i due percorsi.

Rebooting Guest OS & Restarting Applications

Gli hypervisor **monitorano** la **disponibilità del sistema operativo** tramite **heartbeat** guest in una VM, se un guest OS va in crash l'hypervisor cerca di **riavviare il Sistema Operativo**. Se l'**applicazione** va in **crash** ma il sistema operativo no, l'applicazione può essere **riavviata automaticamente (Restarting Applications)**.



Alta disponibilità mediante cluster di nodi fisici

I **cluster virtuali** sono composti da **due o più server fisici** con una risorsa di disco condivisa, quando un **host fisico fallisce** anche tutte le macchine virtuali che dipendono da esso falliscono, se un host **non risponde agli heartbeat** si procede con lo **switch**. L'**unità disco è condivisa** quindi è possibile accedere ai file delle macchine virtuali e **riavviarle in un secondo host fisico**. Esiste anche un meccanismo di **fault tolerance** che permette di **riattivare la macchina e sincronizzarla** allo stesso stato in cui si è spenta ma è **costoso in termini di risorse** e si usa solo in caso di **altissime disponibilità**.

Live migration

Consiste nello **spostamento di una VM** da un **host ad un altro** senza tempi di inattività o **impatti sulle prestazioni**. Applicabile **solo per downtime pianificati**, se un server fisico diventa **indisponibile** non è possibile spostare le macchine virtuali, **necessario spostarle quindi prima del downtime**.

Vantaggioso rispetto **all'ambiente fisico** perché in caso di manutenzione pianificata le applicazioni ospitate dal server **devono andare offline**.

Disaster Recovery

Le macchine virtuali possono essere **copiate e tenute allineate** con tecnologie di **allineamento dei dischi**, le VM potranno essere **istanziate rapidamente** dalle copie disponibili. Una macchina virtuale può essere migrata da una **infrastruttura virtuale on-premise** ad una **infrastruttura basata su cloud senza interruzione**.

Applicazioni IT e sistema virtualizzato

Le caratteristiche rilevanti di un ambiente virtualizzato sono:

- Capacità di CPU riservata ad una VM
- Switch di VM tra host fisici
- Resource Pool

Necessario stabilire un **quantitativo minimo di CPU** che, se l'hypervisor non può allocare, la VM non potrà avviare. In caso di **contesa** o scarsità di risorse la risorsa viene assegnata alla VM in base alla **priorità relativa**.

Se le **risorse** durante l'esecuzione **non** dovessero essere **disponibili** l'hypervisor effettuerà uno **switch a caldo** su un altro **host** che disponga della **risorsa richiesta**. Esistono configurazioni in cui le **risorse** sono **combinare tra loro** e messe a disposizione a più host creando dei **resource pool**.

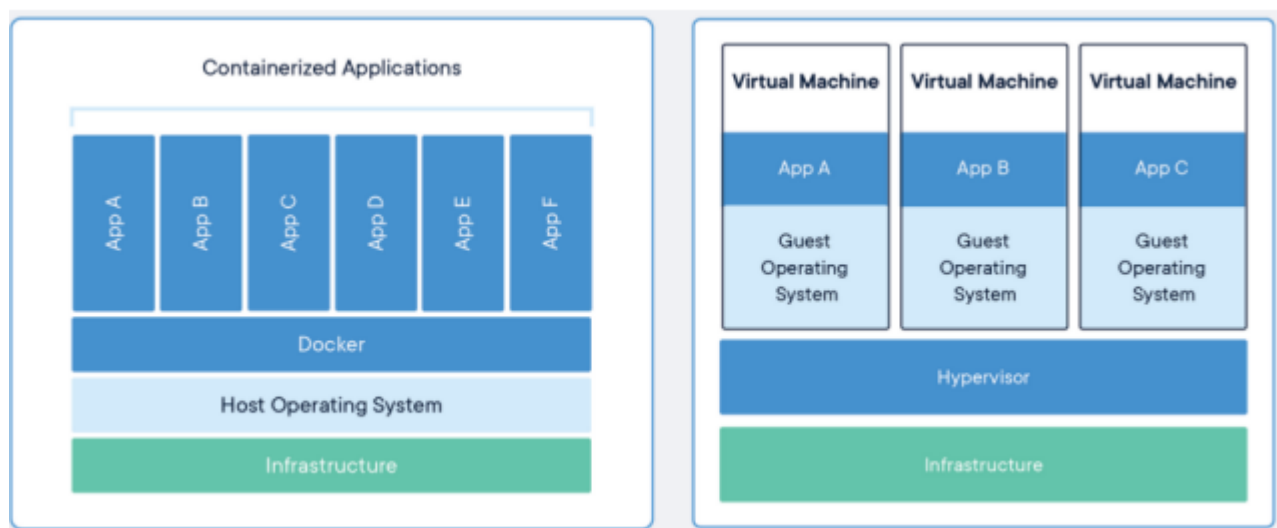
Virtual Appliance

Sono delle **macchine virtuali predefinite** che contengono la **configurazione minima del sistema operativo**, l'**immagine dell'applicativo** e lo **script di configurazione**, sono in formato **OVF** e permettono la portabilità sull'hypervisor. **Contengono tutto il necessario per la loro distribuzione**.

Il sistema operativo chiamato **JeOS** ha **solo ciò che richiede l'applicativo** e non di più, in caso di disponibilità di una nuova versione la macchina virtuale viene sostituita.

Le virtual appliance **riflettono le best practice** per i prodotti che contengono ma nessuna configurazione potrà coprire tutti gli scenari, quando è necessario **aggiornare un modello** può essere fatto tramite **clonazione**.

Cenni sui containers



Rappresentano una tecnologia per **includere le applicazioni** e **tutto ciò** di cui quelle **applicazioni hanno bisogno** per **funzionare**, sono **distribuibili rapidamente** e come le macchine virtuali sono **indipendenti dalla piattaforma fisica** e raggruppano e possono distribuire applicazioni.

I container **condividono il kernel** del sistema operativo con altri contenitori ed **ogni container è isolato nello spazio utente**. Hanno il vantaggio di **occupare meno spazio** delle macchine virtuali e **gestire più applicazioni**. Tuttavia, **restringono l'uso ad un solo sistema operativo**, le applicazioni condividono il kernel.

Sono ampiamente utilizzati in ambienti di sviluppo perché ne consentono una istanziatura multipla e rapida supportando anche la migrazione dell'applicazione tra ambiente on-premise e cloud (**lift & shift**)

Kubernetes

Fornisce un **framework** per eseguire container, si occupa di **scaling** e **failover** nonché dei seguenti servizi:

- **Service discovery e load balancing:** In caso di traffico elevato Kubernetes **bilancia il carico ed il traffico di rete**.
- **Storage Orchestration:** Consente di montare automaticamente un sistema di storage a scelta.
- **Automated rollout and rollbacks:** Possibile descrivere lo stato desiderato per i container distribuiti, automatizzando Kubernetes per creare nuovi container da distribuire.
- **Automatic binary packing:** Fornisce un cluster di nodi a Kubernetes per eseguire container e permette di definire quantità di CPU e RAM necessaria, adatta i container ai nodi per sfruttare al meglio le risorse.
- **Self-healing:** Permette la riparazione automatica ed il riavvio dei contenitori sostituendo quelli che non rispondono al controllo di integrità definito dall'utente.
- **Sensitive Information Management:** Archivia e gestisce informazioni riservate distribuendo ed aggiornando i secret e la configurazione senza ricostruire le immagini ed esporre i secret nella configurazione dello stack.

Cloud Computing

Caratteristiche del cloud computing

Il cloud computing gode delle seguenti **caratteristiche**:

- **Self-service su richiesta:** Cliente acquisisce risorse senza richiedere interazione umana con il provider.
- **Ampio accesso in rete:** Risorse accessibili mediante ogni dispositivo.
- **Servizio misurato:** Ottimizzazione automatica delle risorse.
- **Elasticità:** Risorse scalate in relazione alla domanda.

Deployment Model

Il cloud computing gode di **3 modelli di sviluppo**:

- Pubblico
- Privato
- Ibrido

In un ambiente **pubblico** l'utente paga per l'utilizzo delle risorse insieme ad altri clienti con i quali **condivide le risorse**. Hardware, Software ed Infrastrutture sono **gestite dal provider dei servizi cloud**. Si dice anche ambiente **multi-tenant** in quanto ogni utente si **illude** di avere delle **risorse** dedicate a proprio uso esclusivo.

Vantaggi: costi ridotti, nessuna manutenzione, scalabilità illimitata, alta affidabilità.

In un ambiente **privato** le risorse cloud sono usate esclusivamente dall'utente finale e può essere **situato** nei **data center locali** di chi lo richiede. Hardware, Software ed Infrastrutture sono **gestite in modo privato**.

Vantaggi: Maggiore controllo delle risorse e sicurezza, scalabilità e flessibilità.

Svantaggi: Assenza di elasticità, pooling di risorse e pagamento a consumo.

In un ambiente **ibrido** si hanno risorse sia in cloud **pubblico** che **privato**. Il **cloud bursting** è un modello di distribuzione del carico o di un'applicazione che viene eseguita in un cloud o data center privato e viene 'girata' in un cloud pubblico quando la domanda di **capacità di elaborazione** supera la possibilità del cloud privato.

Vantaggi: Controllo, Flessibilità, Convenienza e Semplicità.

Deployment Model: Benefici e svantaggi

Il cloud pubblico ha **vantaggi** come:

- **Pricing:** L'utente paga solo le risorse consumate
- **Elasticità:** L'utente ha a disposizione un pool di risorse quasi infinito
- **Core competency:** L'utente spende meno tempo alla gestione dell'infrastruttura.

Ed i seguenti **svantaggi**:

- **Riduzione del controllo** da parte del cliente
- **Limitazioni legislative** sull'utilizzo del cloud
- **Mancanza di offerta** necessaria da parte dei fornitori

Cloud Service Models: IaaS, Paas, Saas

Modello		Livello Cloud	Componente		Responsabilità		
SaaS	PaaS	User	Login		Customer	Customer	Customer
			Registration				
			Administration				
		Application	Authentication	Authorization		Provider	
			User Interface	Transactions			
			Reports	Dashboards			
	Application stack	OS	Programming Language	Provider			
		App Server	Middleware				
		Database	Monitoring				
	IaaS	Infrastructure	Data Center	Data Storage	Provider		Provider
Servers			Firewall				
Network			Load Balancer				

Infrastructure as a Service (IaaS)

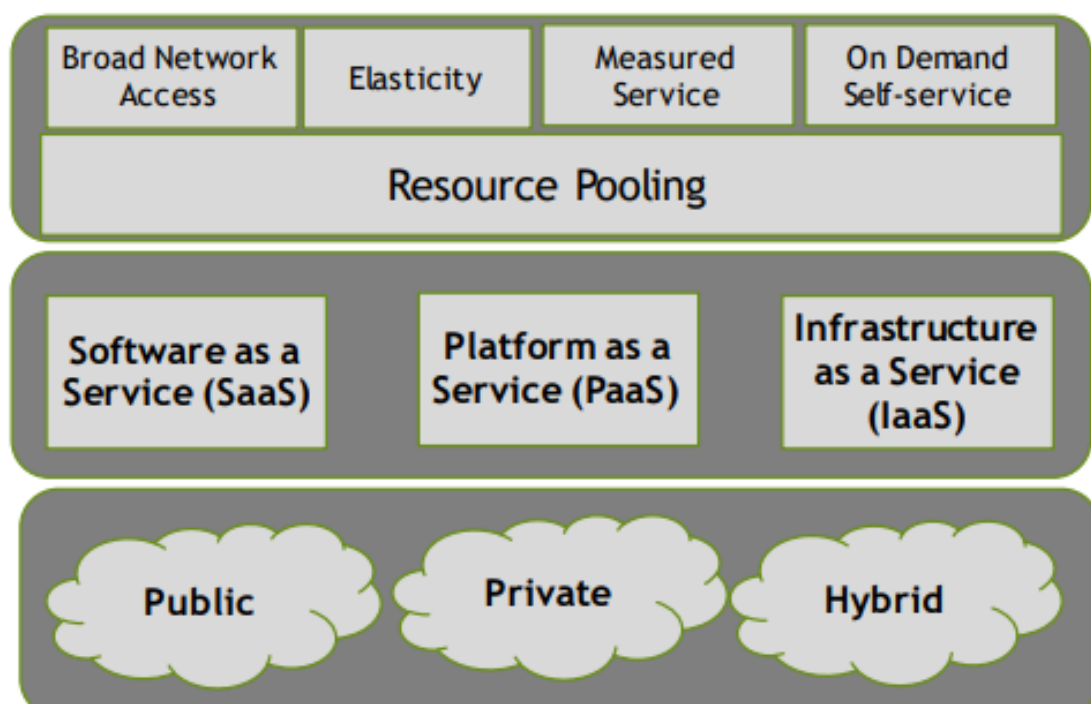
Fornisce al cliente **risorse di elaborazione, archiviazione, reti** ed altre risorse informatiche. Il cliente non gestisce l'infrastruttura cloud sottostante ma ha **pieno controllo** su sistemi operativi, archiviazione ed applicazioni distribuite ed alcune volte ha il controllo anche delle componenti di rete.

Platform as a Service (PaaS)

Fornisce al cliente la **possibilità di implementare** nell'infrastruttura cloud **applicazioni create o acquisite dal cliente** usando linguaggi di programmazione supportati dal provider. Il cliente non gestisce l'infrastruttura cloud sottostante ma ha il **controllo delle applicazioni** e delle **configurazioni** dell'ambiente di hosting.

Software as a Service (SaaS)

Fornisce al cliente la **possibilità di utilizzare le applicazioni del provider** su una infrastruttura cloud e ne garantisce **l'accesso da più dispositivi**. Il cliente non gestisce l'infrastruttura cloud sottostante e **non ha alcun controllo** sulle singole funzionalità dell'applicazione a meno di alcune impostazioni di configurazione **limitate**.



Cloud Service Models

Se un'applicazione ha requisiti di **prestazioni** o **scalabilità** che richiedono **gestione della memoria** e la configurazione dei server applicativi o la **manipolazione del sistema operativo** è preferibile sfruttare **IaaS**, se questi elementi non sono rilevanti allora conviene considerare **PaaS**. **SaaS** permette solo di esternalizzare applicazioni funzionalità e servizi non agevolmente sviluppabili in house.

Infrastruttura Cloud tipica

Il cloud provider suddivide la propria area geografica di copertura in **regioni**. Ogni regione ha centri di calcolo separati (**data center**) dentro i quali ci sono gruppi di **risorse indipendenti** e separate detti domini (**domains**). Ogni dominio è **completamente isolato** dagli altri eliminando così i **Single Point Of Failure**. I collegamenti di rete sono **ridondanti** e ad **altissima velocità** per garantire massima disponibilità in caso di guasti o malfunzionamenti.

Networking in Cloud

Una **Virtual Cloud Network (VCN)** è una rete virtuale privata senza indirizzi pubblici, quindi non routabile su Internet. Le VCN utilizzano **indirizzi IP privati**, identificabili sulla rete locale ma non instradabili pubblicamente.

L'accesso a Internet per una VCN avviene tramite un **Internet Gateway** con indirizzo pubblico che può fungere anche da **firewall**. Le reti private possono estendersi su più data center senza transitare su Internet pubblico, utilizzando gateway di routing dedicati e connessioni sicure come VPN.

Esistono soluzioni di **peering** per **collegare VCN** senza passare per Internet pubblico:

- **Local peering:** collega VCN nello stesso dominio o regione
- **Remote peering:** collega VCN in regioni diverse.

Interazione tra ambienti cloud ed on premise

L'interazione tra sistemi **SaaS** e ambienti **on-premise** avviene in molteplici casi come durante la fase di **caricamento dei dati iniziali**, in fase di **integrazione dei dati di business** e durante **l'esportazione di dati** per analisi e reportistica. Gli applicativi esistenti in una infrastruttura on-premise vengono detti **legacy**, dispongono di sottosistemi dedicati all'import ed export del dato.

Esistono altri tipi di interazioni come invocazioni di **web services** e l'utilizzo di **API**, che consentono l'invocazione di validazione dell'indirizzo e la restituzione di coordinate di geolocalizzazione tramite **REST-API**.

Integrazioni complesse tra ambienti

Integrazione comune a sistemi sviluppati **on-premise** ed in **cloud**, fornisce i dati da elaborare e deve **garantire che i dati siano disponibili** ed attuali coinvolgendo decine o centinaia di sistemi. Può essere **punto-punto** o orchestrata da **middleware**.

Progetto di trasformazione: AS-IS, TO-BE

In un grande progetto informatico si disegna lo **schema delle integrazioni attuali (AS-IS)** e si definisce lo **schema futuro (TO-BE)**. Ogni flusso di integrazione è numerato univocamente, elencato in una tabella con descrizione di sistema sorgente, destinazione, frequenza e volumi.

Nella fase AS-IS si censiscono tutte le interfacce allo **stato attuale**, si passa poi al disegno dell'**architettura futura (TO-BE)** eliminando i flussi superati dalla nuova architettura.

Integrazioni complesse tra ambienti: Middleware

Rappresenta un **software di mezzo** che le applicazioni utilizzano per comunicare tra loro, funge da **ponte tra diverse tecnologie**, strumenti e database. Astrae il processo di comunicazione sottostante le componenti. Fornisce funzionalità per **connettere le applicazioni in modo efficiente**, senza middleware sarebbe necessario costruire un modulo di scambio dati per ogni componente software che si connette alle applicazioni.

Storage in cloud

Esistono 4 tipologie differenti per effettuare lo storage in cloud:

- **Block Storage**
- **File Storage**
- **Object Storage**
- **Local file**

Block Storage Volume

Disco virtuale che fornisce la funzione di **memorizzazione permanente del dato** alle istanze in cloud. I dati vengono allocati in **blocchi di dimensione fissa** che il sistema operativo vede come **mounted volume**, sono spesso usati nelle SAN e **non prevedono la memorizzazione dei metadati**. Supportano qualsiasi tipo di file system. Questa configurazione non esclude la corretta procedura di effettuare **backup regolari** ed effettuare test di restore, una manovra errata si **propagherà su tutti i dischi** rendendo **impossibile ripristinare il dato originale**.

I **block volume** possono essere:

- **Boot Volume**: Dischi per il sistema operativo
- **Data Volume**: Dischi dati

File Storage

Raccolta gerarchica di documenti organizzati in directory che sono a loro volta file strutturati. Devono supportare protocolli di file distribuiti come **Network File System (NFS)**. Si applica a sistemi **general purpose** come aziende dove le unità di archiviazione remota possono effettuare archiviazione comune nei progetti, ma anche in contesti di **supercomputing, Big data & analytics, backup, business continuity, disaster recovery, microservices** e **Docker**.

Object Storage

Tutti i dati vengono gestiti come oggetti. Gli oggetti sono memorizzati in contenitori logici detti **bucket**. Ogni oggetto è composto **dall'oggetto** stesso e dai **metadati** dell'oggetto rendendo più semplice **l'indicizzazione** e **l'accesso ai dati**. Questa tecnologia viene implementata per memorizzare dati non strutturati ed è abbastanza comune perché **altamente scalabile**.

Ogni **oggetto** è **composto da**:

- **Dati**
- **Metadati**: Cosa sono i dati, come vengono usati e la loro riservatezza.
- **Identificatore univoco globale**: Valore a 128 bit non è necessario conoscere la sua posizione fisica.

Storage Gateway

Lo storage gateway è un **gateway di archiviazione cloud** che consente di **connettere le applicazioni locali** allo storage in **cloud IaaS**, associa i propri file e directory agli oggetti con gli stessi nomi in un **bucket in cloud** sull'object storage corrispondente. **Scelta ottimale** per archiviazione dei dati sul cloud permette un'espansione in modo **praticamente illimitato**.

Local File

File memorizzato su un dispositivo locale dell'istanza computazionale è progettato per **applicazioni ad alte prestazioni, l'accesso a blocchi** come nel Block Storage.

Cloud computing – Considerazioni sui costi

Costi e benefici si distribuiscono nel seguente modo:

- **Dipartimento IT**
- **Bilanciamento cash flow**: CAPEX e OPEX
- **Formazione del personale**

Considerazioni sui costi: Dipartimento IT

La variazione del costo è data dalla **diversa ripartizione dei costi** di acquisto tra **hardware e software**, implementazione del sistema e costi di esercizio.

Nei sistemi **on premise** si devono affrontare costi relativi ad **acquisizione di licenze hardware e software, test** e rilascio in produzione. Inoltre, si applicano i **costi ricorrenti di esercizio** relativi alla manutenzione evolutiva del software, gli aggiornamenti delle versioni, **costi dell'energia**, etc...

Nei **sistemi cloud** i costi di acquisizione sono in larga parte trasformati in costi di esercizio (**OPEX**) rimangono i costi di implementazione del software ed il **training**.

DevOps è una metodologia di sviluppo software che punta alla **comunicazione, collaborazione ed integrazione** tra sviluppatori addetti alle Operations, si differenzia da **Agile** che viene visto come un modo per **colmare lacune di comunicazione** tra clienti e sviluppatori, **DevOps** risolve i **divari** tra **sviluppatori** ed **Operations**.

Considerazioni sui costi: Bilancio e cash flow

Il flusso di cassa o **cash flow** è la differenza **tra quantità di denaro in entrata e quantità di denaro in uscita**.

Nel modello **on premise** le licenze vengono **acquistate in anticipo**, ci sono anche da sostenere costi relativi alla **manutenzione annuale** e costi per servizi professionali ed **installazione del software**. Questo grande investimento iniziale ha un impatto negativo sul cash flow, ci vorrà del tempo prima che le entrate coprano le **spese iniziali in conto capitale**.

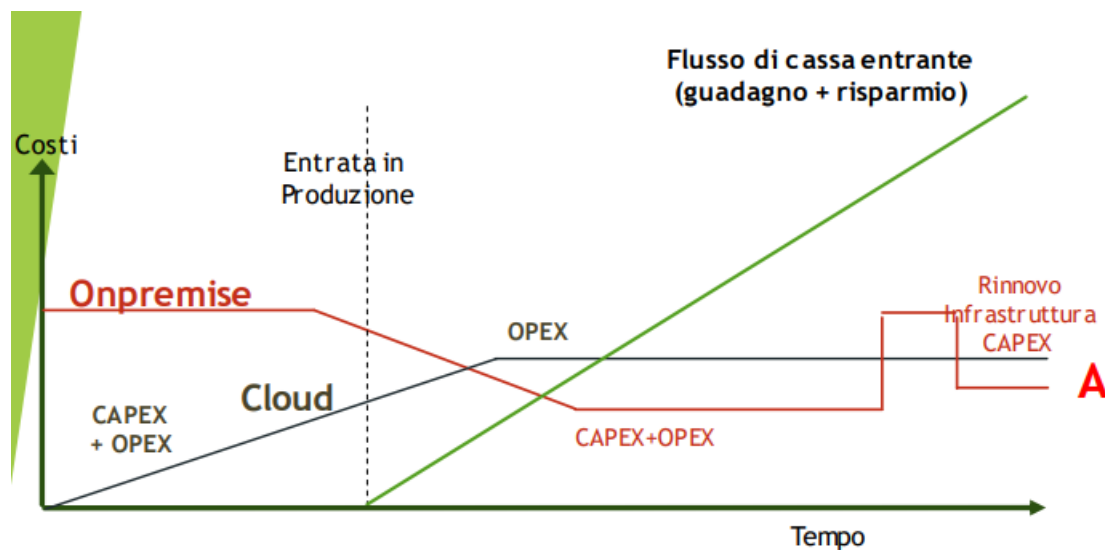
Nel modello **Cloud** spesso si usa la **modalità pay-as-you-go** in cui gli acquirenti **non hanno costi iniziali** e **pagano solo i servizi che utilizzano**. A differenza dei modelli on premise **non ci sono grandi investimenti iniziali** o licenze software da acquistare prima dell'entrata in produzione. **Possibile bilanciare costi e ricavi** liberando il capitale circolante da investire in altre aree del business.

Il Total Cost of Ownership (**TCO**) è composto da due componenti, una fissa ed una variabile, è la somma delle spese di **capitale (CAPEX)** e dei costi **operativi (OPEX)**.

$$\text{TCO} = \text{CAPEX} + \text{OPEX}$$

Il **Return of Investment (ROI)** è il periodo di tempo entro il quale l'investimento viene ripagato dai risparmi che il bene o servizio produce con il suo uso.

Andamento del cash flow



Nel **modello Cloud** il costo totale delle licenze sarà sostenuto quando il sistema andrà in produzione, non si possono escludere a priori dei CAPEX anche se **minori del modello on premise**.

Nel **modello on premise** bisogna considerare i costi relativi al conto capitale CAPEX al quale aggiungere la spesa operativa OPEX, durante la vita del sistema si affronteranno nuovamente ulteriori CAPEX per il rinnovo dell'infrastruttura.

Andamento del cash flow

Il **MOL o EBITDA** è un valore che permette di capire se **un'azienda è in grado di generare ricchezza** tramite la gestione operativa, rappresenta l'autofinanziamento potenziale che identifica il flusso potenziale che l'impresa originerebbe se tutti i ricavi fossero stati riscossi e tutti i costi pagati nell'anno.

Il **MOL** si ottiene sottraendo il **Valore Aggiunto** i costi del personale ed altri costi variabili quali materiali e servizi acquistati.

$$\text{ValoreAggiunto} = \text{ValoreProduzione} - \text{CostiEsterniProduzione}$$

$$\text{MOL} = \text{ValoreAggiunto} - \text{CostoPersonale} - \text{CostiEsterniProduzione}$$

$$\text{MargineOperativoNetto} = \text{MOL} - \text{Ammortamenti} - \text{Accantonamenti}$$

Service Level Agreement (SLA)

Accordo contrattuale sul livello di servizio tra **fornitore** (CSP) e **cliente** (CSC) di servizi cloud. Include un **impegno oneroso da parte del provider**, se il livello non viene raggiunto, il provider dovrà pagare una penale. Gli SLA sono fondamentali i clienti devono essere certi che il provider fornirà servizi **affidabili, sicuri, scalabili e disponibili**. Più è **stringente** un SLA, più **costa** al provider **gestirlo e mantenerlo**.

Definizioni e misura dello SLA

Un contratto di SLA include solitamente la **definizione degli aspetti coperti** dallo SLA ed i **criteri di misurazione** e livelli. I **livelli** definiscono il valore della penale che il provider deve **pagare in percentuale sul contratto** se non viene raggiunto il valore percentuale dello SLA.

- Livello di Raggiungimento compreso tra 99% e 99.5: penale del 10%
- Livello di Raggiungimento compreso tra 98.5% e 98.99: penale del 25%
- Livello di Raggiungimento minore del 98.5: penale del 100%

Elasticità e scalabilità

Una delle caratteristiche del cloud computing è che **il fornitore** di servizi cloud (CSP) deve poter **rispondere alle esigenze del cliente** in termini di elasticità. Il CSP utilizza tutte le tecniche per **garantire scalabilità**.

La **virtualizzazione garantisce** per il cliente: **variazione dinamica delle risorse**, la possibilità di **istanziare nuove macchine virtuali**, **alta disponibilità** e **failover** per caduta di macchine virtuali o nodi fisici.

Criteri generali di **scalabilità e disponibilità riguardano: cluster** di server fisici in load balancing, dischi in alta affidabilità come **RAID, SAN** e **NAS** e possibilità di **scalabilità verticale**.

Autoscaling

L'autoscaling è l'**adeguamento automatico della capacità elaborativa** in IaaS disponibile presso fornitori cloud.

Può essere:

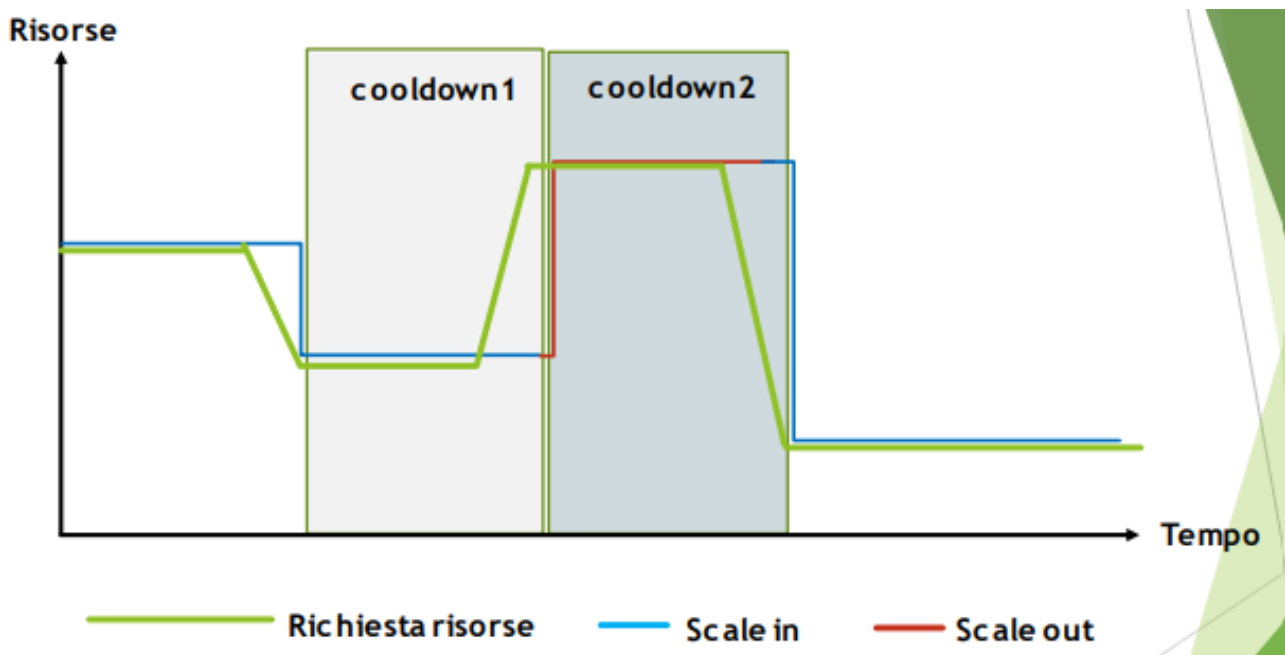
- **Automatico**: Al raggiungimento di una certa soglia
- **Pianificata**: In funzione di periodi di picco ricorrenti

Si utilizza per gestire automaticamente le dimensioni del pool di istanze, si esegue il provisioning ed il pool viene **aumentato (scale out)** o **ridotto (scale in)**.

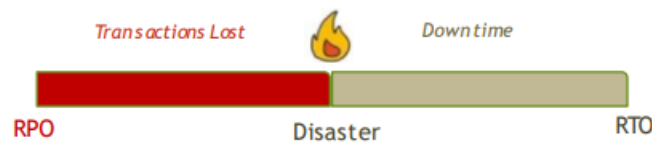
Nella scalabilità automatica, si sceglie una **metrica da monitorare** e si impostano delle **soglie** che, se raggiunte **dimensionano automaticamente il pool di risorse in tempo reale**, all'aumentare del carico il pool cresce e quando diminuisce si ridimensiona.

Il **cooldown** è un **periodo di stabilizzazione** tra gli eventi di scalabilità automatica. Il periodo **inizia quando il pool raggiunge il nuovo livello richiesto**, solo al termine del cooldown l'autoscaling potrà modificare nuovamente le dimensioni del pool se necessario.

Al raggiungimento della soglia l'autoscaling effettua il **provisioning** (scale out) o **riduzione** (scale in) solo se è passato il periodo di cooldown dell'adeguamento precedente.



Continuità Operativa: RTO e RPO



- Il **Recovery Point Objective (RPO)** è la massima perdita di dati ammissibile.
- Il **Recovery Time Objective (RTO)** è l'intervallo di tempo massimo per il recovery.

Il **Recovery Time Objective** è il **tempo** entro il quale l'azienda **richiede che il servizio sia di nuovo attivo** e funzionante.

Il **Recovery Point Objective** è l'obiettivo del punto di ripristino o la **quantità di dati che può essere tollerata**.

Il **valore di ripristino** misura quanto vale per l'azienda **mitigare le situazioni di disastro**, ci sono molti fattori che possono influenzare questo valore come la criticità del servizio, se un servizio è fondamentale o vitale avrà un valore di ripristino molto elevato.

Continuità Operativa: Disaster Recovery in cloud

Strategie di **continuità operativa** e **disaster recovery** identiche a quelle dei sistemi **on premise**. Relativamente alle tecniche di disaster recovery i grandi provider hanno **data center in più nazioni** e **continenti** differenti. Questa **dislocazione** garantisce il **failover** e la localizzazione del dato. Nei cloud il **ripristino di emergenza** **incapsula** l'intero server, il sistema operativo, le applicazioni, le patch ed i dati **in un singolo server virtuale**. Questo server virtuale viene copiato e sottoposto a backup, i dati possono essere **migrati** da un data center ad un altro in **modo molto più veloce** rispetto ai tradizionali approcci di disaster recovery.

Disaster Recovery – Opzioni in cloud

Nel **dynamic provisioning** le risorse vengono attivate al momento dell'evento disastroso. Questo garantisce **costi più bassi** ma lentezza nel reagire e quindi un **recovery time objective più elevato**.

Nel **pre-provisioned at reduced footprint** le risorse sono in parte attive e le rimanenti sono fornite al momento dell'evento disastroso. Garantisce **costi relativamente bassi** ma una lentezza nel reagire con un **RTO ancora più lungo**.

Nel **pre-provisioned images kept offline** le risorse sono fornite integralmente ma tenute offline. L'ambiente viene avviato al momento del disastro. **Costi** paragonabili al **precedente** ma **RTO inferiore**.

Nel **pre-provisioned production equivalent** la situazione è paragonabile ad un Disaster Recovery on premise, tutte le macchine sono disponibili e viene **minimizzato il tempo di RTO**.

Option	Tradeoff
Dynamic Provisioning <ul style="list-style-type: none">• Target VMs NOT pre-provisioned• Provisioned at DR event	Lowest cost Longest RTO
Pre-Provisioned at Reduced Footprint <ul style="list-style-type: none">• Pre-provision some/all VMs at smaller HW spec	Medium cost Better RTO
Pre-Provisioned – Images Kept Offline <ul style="list-style-type: none">• Sync'ing storage in VM• Does <i>not</i> incur OS licensing costs	Medium cost Even Better RTO
Pre-Provisioned – Production Equivalent Pre-provision some/all VMs at full HW spec	Highest cost Best RTO

Ambienti e cicli di sviluppo

La prima fornitura di ambienti da parte del provider è detto **provisioning**. Durante lo sviluppo di un applicativo, sono forniti almeno tre ambienti: **sviluppo**, **test** e **produzione**.

Gli ambienti di sviluppo garantiscono **elasticità** in quanto il **provisioning** è **garantito da ulteriori ambienti** durante il ciclo di vita del sistema. Gli ambienti possono essere **forniti** e **dismessi** offrendo grande **flessibilità** rispetto al ciclo di sviluppo on premise.

In un ambiente **virtuale** è anche possibile **richiedere per un periodo di tempo** limitati un **ambiente di caratteristiche vicine** o uguali alla produzione per eseguire test.

Update automatici: P2T e T2T

Il cliente **accetta da contratto gli update del software** resi disponibile dal fornitore cloud, gli **ambienti** vengono **aggiornati in sequenza**, prima in un ambiente di test per poi procedere con l'upgrade in produzione. Si applicano processi indicati come:

- **Production To Test (P2T)**
- **Test To Test (T2T)**

Gli upgrade potrebbero non richiedere downtime, eventualmente **data e tempo di downtime saranno comunicati con largo anticipo** dal fornitore ed il cliente deve sempre eseguire un **no-regression-test** per verificare che le funzionalità di un sistema operativo non siano state danneggiate.

Protezione del cloud

- **Bastion Host**

- Istanza computazionale nel cloud IaaS.
- **Punto di ingresso sicuro e controllato** alla rete dall'esterno.
- Installato in una zona demilitarizzata (**DMZ**).
- Protegge **risorse critiche** in reti private inaccessibili dall'esterno.
- **Unico punto di ingresso** monitorabile e controllabile.

- **Network Security Group (NSG)**

- Firewall virtuale per risorse cloud con lo stesso livello di sicurezza.
- Include VNIC (Virtual Network Interface Cards).
- Controlla il traffico in entrata e in uscita tramite regole di sicurezza.

- **Security List**

- **Firewall virtuale per un'istanza.**
- Specifica i tipi di traffico consentiti in entrata e in uscita.
- **Applicata a livello di sottorete**, tutte le VNIC in una sottorete sono soggette alle stesse regole.
- Definisce **regole di sicurezza** per tutte le VNIC di una sottorete specifica.