Lezione introduttiva su Architetture e Infrastrutture Hardware/Software

Quando parliamo di *architetture* e *infrastrutture* in ambito informatico, ci riferiamo a due concetti strettamente collegati ma distinti:

1. **Architettura (Architecture)**: riguarda la progettazione logica di un sistema (hardware e/o software), ovvero come sono organizzate le sue componenti e quali sono le relazioni tra di esse.
2. **Infrastruttura (Infrastructure)**: comprende tutti i componenti fisici e virtuali (server, reti, dispositivi di storage, sistemi operativi, servizi cloud, ecc.) che permettono al software di funzionare correttamente.

Vediamo più da vicino i concetti chiave:

---

1. Architettura Hardware

1.1 Componenti fondamentali

- **CPU (Central Processing Unit)**: è il “cervello” del sistema. Si occupa dell’esecuzione delle istruzioni e del controllo delle operazioni.
- **Memoria (RAM)**: memorizza i dati e le istruzioni in uso in un determinato momento, garantendo un accesso rapido alla CPU.
- **Memoria di massa (Storage)**: comprende dischi rigidi (HDD) o unità a stato solido (SSD), dove risiedono sistemi operativi, programmi e dati a lungo termine.
- **Bus e schede di espansione**: collegano CPU, RAM, dispositivi di input/output (I/O), rete, ecc., permettendo il transito dei dati all’interno del sistema.
- **Periferiche (I/O)**: mouse, tastiera, monitor, stampanti, dispositivi di rete, ecc.

1.2 Tipologie di architettura

- **Architettura Von Neumann**: un singolo sistema di memoria per dati e istruzioni; è il modello di riferimento più diffuso nei computer general-purpose.
- **Architettura Harvard**: separa la memoria delle istruzioni da quella dei dati, tipica di alcuni microcontrollori.
- **Architetture parallele e multicore**: introdotte per migliorare le prestazioni, consentono l’esecuzione simultanea di più thread/processi.

---

2. Architettura Software

2.1 Concetti di base

L’architettura software descrive come i componenti di un’applicazione (o di un sistema più grande) sono organizzati e come interagiscono. Una buona architettura deve essere:

- **Manutenibile**: semplice da aggiornare o correggere.

- **Scalabile**: capace di gestire un aumento di utenti o dati senza perdere in prestazioni.
- **Affidabile**: tollerante ai guasti o ai malfunzionamenti parziali.
- **Sicura**: deve tenere conto di aspetti di sicurezza e protezione dei dati.

2.2 Pattern architetturali comuni

- **Monolitico**: tutte le funzionalità sono integrate in un unico blocco software. Facile da sviluppare inizialmente, ma può diventare difficile da mantenere e scalare.
- **Client-Server**: il client (ad esempio un'applicazione desktop o un browser) richiede servizi e il server elabora e risponde. È il modello base di gran parte delle applicazioni web.
- **Layered (a livelli)**: l'applicazione è suddivisa in livelli (presentation, business logic, data access, etc.). Ogni livello interagisce con quello sottostante. Favorisce la separazione dei compiti.
- **Microservizi**: l'applicazione è divisa in tanti servizi piccoli, indipendenti e specializzati che comunicano tra loro (spesso via API REST). Favorisce l'agilità e la scalabilità, ma aumenta la complessità infrastrutturale.
- **Event-Driven**: i componenti reagiscono a eventi generati da altri componenti (pub/sub). Ideale per sistemi distribuiti e scalabili.

---

3. Infrastrutture Hardware/Software

3.1 Infrastruttura tradizionale (On-Premise)

- **Server fisici**: macchine collocate in un datacenter aziendale o in un locale dedicato (server room).
- **Reti locali (LAN)** e dispositivi di networking (router, switch, firewall).
- **Sistemi di backup e storage dedicati**: NAS, SAN, dischi esterni, ecc.
- **Sistemi operativi e ambienti di virtualizzazione**: es. VMware, Hyper-V per consolidare più macchine virtuali (VM) su un singolo server fisico.

3.2 Virtualizzazione e Containerizzazione

- **Virtualizzazione**: permette di eseguire più sistemi operativi (VM) su un unico host. Ogni VM è isolata dalle altre e dispone di risorse virtuali (CPU, RAM, storage, rete).
- **Containerizzazione** (ad es. Docker): consente di eseguire applicazioni all'interno di container leggeri che condividono lo stesso kernel del sistema operativo, ma sono isolati in termini di processi, rete e filesystem. Riduce overhead e semplifica il deployment.

3.3 Infrastrutture Cloud

- **IaaS (Infrastructure as a Service)**: fornitura di risorse virtuali (VM, rete, storage) come servizio, ad esempio AWS EC2, Azure Virtual Machines, Google Compute Engine.
- **PaaS (Platform as a Service)**: fornitura di una piattaforma di sviluppo e deployment, ad esempio AWS Elastic Beanstalk, Heroku, Google App Engine.
- **SaaS (Software as a Service)**: applicazioni pronte all'uso fornite via web, ad esempio Gmail, Office 365, Salesforce.

- **Vantaggi del Cloud**: scalabilità elastica, pagamento a consumo (pay-per-use), maggiore agilità e minor tempo di provisioning.
- **Svantaggi**: dipendenza dalla connettività di rete, possibili preoccupazioni di sicurezza e conformità, lock-in del fornitore.

3.4 Infrastrutture Ibride

Molte aziende adottano soluzioni ibride, combinando:

- **Cloud Pubblico** (AWS, Azure, GCP) per servizi scalabili e dinamici.
- **Datacenter privati** (On-Premise) per workload critici, dati sensibili o applicazioni legacy.

---

4. Scalabilità e Alta Disponibilità

4.1 Scalabilità

- **Verticale (Scale-up)**: aggiungere risorse hardware a un singolo server (es. più CPU, RAM). Ha un limite fisico ed economico.
- **Orizzontale (Scale-out)**: aggiungere più server (nodi) al sistema, condividendo il carico (load balancing). Consente maggiore flessibilità e affidabilità.

4.2 Alta Disponibilità (High Availability)

- **Ridondanza**: duplicare componenti o servizi per gestire eventuali guasti (es. server in cluster).
- **Failover**: passaggio automatico da un sistema primario a uno secondario in caso di malfunzionamento.
- **Load Balancing**: bilanciare il carico tra più istanze di un servizio, riducendo i colli di bottiglia e aumentando la disponibilità.

---

5. Sicurezza e Monitoraggio

5.1 Sicurezza

- **Firewall, VPN, segmentazione di rete**: controllano e filtrano il traffico in entrata/uscita, limitando l’accesso non autorizzato.
- **Sicurezza applicativa**: proteggere da SQL injection, XSS, CSRF, ecc. tramite best practice di sviluppo e sistemi di protezione (WAF - Web Application Firewall).
- **Autenticazione e Autorizzazione**: gestire credenziali, privilegi e diritti di accesso per utenti e servizi.

5.2 Monitoraggio

- **Strumenti di logging** (ad es. ELK stack: Elasticsearch, Logstash, Kibana) per raccogliere e analizzare i log di sistema.

- **APM (Application Performance Monitoring)**: New Relic, Datadog, Grafana, Prometheus, ecc. per monitorare metriche di performance e rilevare problemi in tempo reale.
- **Alerting**: impostare soglie e notifiche per agire tempestivamente su anomalie.

Conclusione

Un'architettura ben progettata e supportata da un'infrastruttura adeguata è fondamentale per garantire le prestazioni, la scalabilità e l'affidabilità di un sistema informatico. La scelta dipende da molte variabili: requisiti di business, costi, sicurezza, conformità, prestazioni richieste, quantità e tipologia dei dati, ecc.

I passaggi chiave per costruire e gestire un sistema sono:

1. **Analisi dei requisiti**: comprendere a fondo gli obiettivi (funzionalità, numero di utenti, volumi di dati, SLA richiesti).
2. **Progettazione dell'architettura** (hardware e software): selezionare pattern e tecnologie più adatti al contesto.
3. **Scelta dell'infrastruttura** (on-premise, cloud, ibrida, container, microservizi, ecc.) in base a costi, competenze e obiettivi di scalabilità.
4. **Implementazione e test**: assicurarsi che il sistema rispetti i requisiti funzionali e di performance.
5. **Monitoraggio continuo e gestione**: per mantenere la stabilità, reagire ai guasti e pianificare in anticipo l'aumento di risorse.

In sintesi, l'informatica moderna ha a disposizione molteplici modelli architetturali e infrastrutturali. Comprendere le differenze e saper scegliere (o combinare) le soluzioni più adatte consente di costruire sistemi robusti, performanti e facilmente manutenibili.