**LEZIONE 3: DESIGN**

**Modello Broadcast**

Nel mezzo canale che fa mediatore, il quale decide a chi inviare i messaggi (Broadcasting selettivo, vedo solo messaggi che mi interessano)

**ARCHITETTURE A SISTEMI DISTRIBUITI**

**Vantaggi:** Distribuendo il carico, fare più cose in maniera più veloce (Complessità)

**Quando distribuisco pero,** la gestione della concorrenza diventa un problema, il tutto è più difficile da maneggiare e più impredibicibile (a causa dei molteplici processi che lavorano)

**Architettura client->server:** Servizi distribuiti chiamati dai client, i server chiamati in causa vengono usati in modo differente dal client chiamante, ricorda che client e server non devono essere visti come dispositivi singoli, ma come software in generale (Noi possiamo avere più computer che fungono da server paralleli per quel sistema)

**Architettura oggetti distribuiti**: No distinzione tra client e server, ogni oggetto sul sistema potrebbe usare servizi da altri oggetti

**Architettura multiprocessor:** Sistema costituito da processi multipli che possono operare su diversi processori

**Architettura applicazioni stratificata:**

1. **Livello di presentazione:** Si occupa di presentare i risultati di diverse computazioni dell’utente del sistema e collezionare gli input dell’utente
2. **Livello di elaborazione delle applicazioni**: Si occupa di fornire funzionalità specifiche per l’applicazione **(In un sistema di una banca, funzioni bancarie come apertura di un account, chiudere un account ecc..)**
3. **Livello Data management:** Si occupa di gestire il database di sistema

**Bisogna differenziare tra due tipi di applicazione stratificata**

**Client Magro – Client Grosso**

* **CM**: in questo modello, tutta l’elaborazione delle applicazioni e data management vengono eseguite sul server. Il client è responsabile solo di runnare il sfotware di presentazione
* **CG**: Qui il server è solo responsabile del data management, il sotware sul client implementa la logica dell’applicazione e le interazione con l’utente del sistema

**CLIENT MAGRO**: Usato quando sistemi legacy (?) vengono migrati ad un’architettura client server

* + Sistemi Legacy: agiscono come un server a sé stante con un’interfaccia grafica implementata su un client
  + Svantaggio maggiore è che il carico di elaborazione è elevato sia sul server che sul network

**CLIENT GROSSO:**

* + Una **maggiore quantità di elaborazione** viene delegata al client, poiché l’elabiorazione dell’applicazione viene eseguita localmente
  + **Più adatto ai nuovi sistemi C/S**, dove le capacità del sistema del client sono conosciute in anticipo
  + **Gestione più complessa rispetto ad un CM**, una nuova versione dell’applicazione deve essere installata su tutti i client

**Architettura a tre livelli:**

* + In un’archittetura del genere, ciascuno dei livelli architetturaapplicativa **possono essere eseguiti su processori separati**
  + Dà **performance migliori** rispetto ad un approccio **client magro**, **più facile da maneggiare** rispetto ad un approccio **client grosso**
  + E’ un tipo di **architettura più scalabile**, come la domanda aumenta si possono aggiungere server extra al sistema

**Architettura peer-to-peer**

* Pari a pari, **qui nessuno è client o server**, ma tutti i moduli dell’architettura si comportano sia da client che da server (Molto utile in applicazioni file sharing come emule)
* Ne esistono due tipi: **Decentralizzate e semi-centralizzate**

**Architettura orientata al servizio**

* Le applicazioni si trasformano in servizi web, un web service è un approccio standard per creare componenti riutilizzabili e accessibili attraverso il web.
* Servizi sono basati su XML standars, da poter essere distribuiti su diverse piattaforme e scritte in linguaggi differenti.

**UML(Unified Modeling Language)**

* Si usa per comunicare l’idea del progetto che abbiamo, tramite modelli ecc.., senza la necessità di esporre eventuali dettagli superflui, identificando gli oggetti principali del sistema
* Puoi vederlo come quel qualcosa che ti crea il progetto della casa da esporre al cliente

**Conteso del sistema e modelli d’uso**

* Sviluppo di una relazione comprensibile tra sfotware in fase di progettazione e il suo ambiente esterno

**IDENTIFICAZIONE DEGLI OGGETTI**

* Non c’è una formula per l’indentificazione degli oggetti da mettere nei package, bisgona tenere conto dell’utilità di ogni oggetto
* E’ un processo iterativo che si ripete, è improbabile riuscirci al primo tentativo