Sistemi e Applicazioni di Rete

CRISTIAN MERCADANTE
UNIMORE | ULTIMA MODIFICA: 23/07/2019

Lezione 1

Lunedì 25 febbraio 2019

Esame consiste in una prova in laboratorio, superata la quale si fa l'esame orale. Il venerdì si terranno le lezioni di laboratorio. Portare il portatile con materiale già scaricato. Serve Linux.

Site: https://weblab.ing.unimo.it/sar Username: SAR1819 Password: Saerug0s

Introduzione

La parola chiave è *"informatica come servizio"*. Questa frase banale per gli utenti significa qualcosa, per gli ingegneri qualcos'altro.

- L'**usabilità** è un aspetto importante. L'informatica che introduce un manuale di istruzioni è un'informatica progettata male.
- (Apparentemente) **gratuito**, perché paghiamo con i nostri dati.
- **Sempre accessibile**. Informatica che funziona H24. **Sempre disponibile**, ovunque, da qualunque dispositivo, *indipendentemente dal numero di utenti connessi*.

L'utente vede il servizio, mentre l'ingegnere deve impegnarsi col design, build, test, documenti, monitoraggio, fix, miglioramenti. Ma come vengono erogati questi servizi e dove si trovano? All'utente non importa niente, l'importante è che funzioni. Mentre per l'ingegnere è molto importante.

Dal 1969 è stata creata connettività, collegato aziende, università e case. Ma la connettività iniziale era Telnet, FTP, bisognava conoscere IP (cose da ingegneri). Poi è nato il web, un'interfaccia al di sopra di internet dove ci si può muovere da server in server cliccando semplicemente, con un'interfaccia grafica. È nato il concetto di navigatore, che si muove fra informazioni e non fra server. Abbiamo separato l'uso della rete, dalla consapevolezza (1996). Però a quell'epoca i siti non erano ben dimensionati, perché c'è stata l'esplosione del web. Da allora sono cambiate le informazioni, il commercio elettronico, i social media. Contemporaneamente nasceva il cloud, dove il principio era l'informatica come servizio, alle infrastrutture ci penso io. La maggioranza delle persone affermava fosse solo una moda temporanea, sia il web che il cloud. Amazon è nata come commercio elettronico di libri, è stata l'inventrice del cloud, perché il sito doveva essere sempre online e scalabile. Tutte queste tre tecnologie avevano una visione mondiale. Un applicativo non è rivolto a 100 persone all'interno di un'azienda, ma deve essere rivolto a tutto il mondo. Il sistema deve essere scalabile, ovvero non deve degradare le prestazioni quando dati o utenti aumentano. Bisogna introdurre questo concetto sin dal progetto.

Oggi si dà per scontato che i servizi siano accessibili mediante internet. Questo ha comportato problemi di sicurezza. Inoltre, internet non è controllabile, non ci sono regole. L'utilizzo di internet continua a crescere e la tecnologia non è più un limite (avendo adeguato tempo e denaro). Al tempo d'oggi c'è *zero tolerance*. Si dice che la concorrenza *"is just one click away"*. Oggi abbiamo anche a disposizione un esagerato quantitativo dei dati e informazioni. Noi siamo profilati e le informazioni sono filtrate sulla base dei nostri profili. Bisogna inoltre tenere conto degli obiettivi di business della propria azienda.

Lezione 2

Martedì 26 febbraio 2019

Servizio

Un servizio è l'abilità di un sistema di fornire continuativamente uno o più risposte in presenza di specifiche richieste al sistema dal cliente. Ogni servizio ha chi lo eroga e chi lo utilizza. Che significa con continuità? È legata a due aspetti. Uno è contrattualizzato, ovvero un Service Enable Agreement, ovvero un contratto sulle prestazioni, affidabilità e continuità del servizio stesso. Ci sono ovviamente anche pagamenti e penali. Nei servizi di oggi, possono non esserci contratti formali (es. riproduzione video su YouTube), è utile pensare a queste cose, perché le penali sono nello scarso rendimento dato dal servizio (il cliente non usa il sito, ad esempio). Noi dobbiamo imparare a progettare servizi che garantiscano una qualità indipendente dal numero di utenti, che funzionino sempre (robusti) e che siano sicuri. I servizi possono essere gratuiti o no, ma in ogni caso vogliamo guadagnarci: qual è il business model? L'erogazione di servizi gratuiti ad esempio nasconde profilazione, altri usano pubblicità, ecc. Un ultimo aspetto importante è l'usabilità.

Hanno elevata importanza i **requisiti**. Uno dei problemi nel momento in cui dobbiamo realizzare servizi è avere molto chiari i requisiti. Purtroppo, il committente non sa esattamente quello che vuole. La teoria dice che tutti i requisiti devono essere trovati,

ma nella realtà il cliente ha solo una vaga idea, spesso non è soddisfatto e ci sono incomprensioni. Ci vuole un'idea, i requisiti, i dettagli di progetto e poi la realizzazione. È un'azione prettamente ingegneristica. I requisiti possono essere di tipo funzionale e non funzionali. Quelli funzionali definiscono cosa deve fare il sistema (input, output). Quello che ci interessa è la parte non funzionale, quindi il come questi servizi vengono erogati e con quale qualità. I principi che andremo a perseguire sono le prestazioni, la scalabilità, disponibilità e costi. In tutto ciò abbiamo due nemici: *single points of failure* e *bottlenecks*. L'unico metodo per superare questi problemi è la **replicazione** di macchine, dati e processi.

Posso acquistare dei calcolatori/server. Quindi se sono un'azienda creo il mio data center. Oggi i tempi sono cambiati e si fa outsourcing (esternalizzazione), che si può fare mediante tecniche di *hosting collocation* (non sono possessore delle macchine) oppure *housing* (qualcuno ospita il mio datacenter, con le mie macchine, garantendo elettricità, ecc.). Da 15 anni c'è anche la possibilità di usare come outsourcer un cloud provider. Una terza possibilità (attualmente la più frequente) è la situazione ibrida. Quindi possiamo progettare e costruire, noleggiare o comprare.

Quando pensiamo a un progetto, è necessario individuare il proprio ruolo: possiamo essere committenti di noi stessi (es. startup), possiamo essere dipendenti di un'azienda, oppure il committente è un'azienda esterna, perché la nostra azienda fornisce clienti esterni. L'utilizzatore può essere la nostra azienda, un'altra azienda oppure il mercato. Gli utenti chi sono? Le lingue quali sono? (questione non banale).

Ci sono vari tipi di errori nel momento in cui si definiscono i requisiti. I vincoli di tempo, costo e chiari requisiti sono fondamentali. Devono esserci vincoli funzionali, di Service Level Agreement, legali e di workload. Il **Service Level Agreement** è un documento che punta a specificare il tipo di contratto che si ha col fornitore. Il **workload** è il carico di lavoro che arriva alla mia applicazione, ovvero il numero di richieste e il tipo di richieste che dovrò sostenere per avere un servizio prestante e affidabile. L'altro aspetto importante è l'**architettura**. Il progetto deve essere totalmente agnostico rispetto alla tecnologia. Solo dopo si passa a capire come implementare.

Quando pensiamo a un servizio moderno dobbiamo pensare a quattro **livelli logici**:

- User interface: l'aspetto che avrà il servizio per l'utente
- Presentation logic: la parte di software che eroga le funzionalità
- Business logic: crea le risposte
- Dati.

Abbiamo i livelli logici che devono essere progettati, ci sono i **processi** che costituiscono i livelli logici e infine ci sono i **computer** che eseguiranno i processi. Concettualmente possiamo dire che l'informatica vista finora girava solo su una macchina. È più probabile che ci sia almeno un processo client, almeno uno sulla parte server, oppure più server, ciascuno che implementa logiche diverse. Posso avere tutti i processi che voglio. Per quanto riguarda i computer, posso avere diverse combinazioni a seconda del numero di processi. Ad esempio, è possibile avere i quattro processi logici in esecuzione su quattro computer differenti. Quindi bisogna pensare sia alle architetture logiche che a quelle fisiche (oggi ci sono anche le macchine virtuali).

Quando creiamo un servizio dobbiamo capire se il servizio è statico, di publishing, di e-commerce, di self-service, di trading, di intermediazione, di social-networking, di gaming. Ciascuno di questi servizi ha il proprio workload. I contesti principali sono il **Business-to-Consumer (B2C)**, ovvero chi vende al cliente, oppure c'è il **Business-to-Business (B2B)**, infine ci sono i servizi **Customer-to-Customer (C2C)**, come ad esempio EBay. È importante quindi capire il contesto in cui siamo, e i requisiti che ci dà il contesto.

Oggi il workload è molto variabile, eppure bisogna starci dietro. La cosa interessante è che il profilo di richieste orario è indifferente dalla time-zone. Però le richieste si sommano in ogni istante. Bisogna quindi tenerne conto. Anche la tipologia di servizio ha picchi differenti e unici. Il concetto è che bisogna stare molto attenti al workload e bisogna fare delle previsioni di crescita e sovradimensionare. All'esigenza ci si espande. Ma il vero workload non è lineare ed è imprevedibile: in alcuni momenti stiamo sprecando risorse, in altri momenti non riusciamo a soddisfare l'utente. Ma grazie al cloud, le risorse fisiche possono seguire il profilo di carico vero (e non quello atteso) e quindi non avere perdite e sprechi (è pubblicità, ma il concetto è quello). Introduce elasticità, ovvero la capacità di crescere e di diminuire (è una scalabilità bidirezionale).

Cloud (1)

La vision inizia negli anni 60, parlando di calcolo/computazione come public utilities. Jeff Bezos (CEO di Amazon) nel 2006 ebbe l'idea di usare le risorse inutilizzate per inventare il cloud computing. Per erogare i propri servizi ha dovuto fare enormi investimenti architetturali, ma non sempre sono tutte utilizzate. Quindi ha pensato di dare le proprie infrastrutture in noleggio ad altri. Le public utilities sono l'acqua, la luce, il gas e il telefono. Hanno in comune il pagamento al consumo, la continuità del servizio. Amazon utilizza la stessa piattaforma sia per gestire i siti, sia per offrire processi, server e servizi agli altri.

Il cloud dobbiamo considerarlo come una famiglia con diverse tipologie di servizi e metodi. I tre tipi di servizio sono i principali:

- **Infrastructure as a Service (IaaS)**: è la parte hardware. Tu puoi noleggiare i processori, macchine, dischi, RAM, connessioni fra server, tempo. Il software ce lo metti tu e gestisci tu.
- **Software as a Service (SaaS)**: usi il servizio, ad esempio Gmail, dove non abbiamo idea dell'infrastruttura.
- **Platform as a Service (PaaS)**: è il livello più interessante per gli informatici, si interpone fra i precedenti. Sono le funzioni, le librerie, gli ambienti che possiamo utilizzare per sviluppare la propria applicazione, ambienti di testing, API, ecc.

Il vantaggio è che possiamo fare applicazioni che prima non potevamo fare, facilitando anche lo sviluppo grazie alla disponibilità di librerie. Lo svantaggio è il vendor lock in: siamo bloccati all'utilizzo della platform.

Tutto ciò è soltanto uno specifico modello di deployment, chiamato public cloud. Nulla vieta di crearsi il proprio datacenter secondo la modalità cloud, ottenendo un private cloud. Oppure posso unirmi ad altre aziende e far un community cloud. Esiste anche l'opzione ibrida. Per capire cosa mettere sul cloud e cosa tenere in privato serve un ingegnere che consideri i vari fattori di costo.

Lezione 4

Martedì 2 marzo 2019

Cloud (2)

IaaS, SaaS e PaaS sono solo i paradigmi più diffusi. A seconda delle esigenze sono nate situazioni ibride. I servizi forniti da un cloud provider sono sempre più variegati e complicati. Per capire quale utilizzare e sapersi orientare richiede esperienza e competenze. Le aziende che si stanno buttando sul cloud cercano persone esperte oggigiorno. L'altro aspetto da prendere in considerazione è non focalizzarci solo sul cloud pubblico, ma pensare alle altre scelte: possiamo creare cloud privati. Il privato può voler dire il privato che ci si fa in casa: ad esempio, l'università potrebbe offrire servizi organizzando il suo hardware con una profonda virtualizzazione per offrire servizi di IaaS, SaaS o PaaS. Quindi l'università potrebbe voler offrire servizi ai clienti interni, quindi togliere tutte le macchine dai piccoli datacenter, centralizzare e offrire ai propri clienti/colleghi le tecnologie che ci offre un cloud provider. La stessa tendenza si può estendere in un discorso di comunità. Invece di avere tantissimi piccoli datacenter che non riesco a gestire, ne creo uno che possa offrire servizi a tutte le aziende simili del territorio che hanno le stesse esigenze, creando un community cloud. Oggi è molto probabile che si utilizzi una soluzione ibrida, quindi con alcuni servizi e dati in casa e altri in cloud. Grandi aziende multinazionali (es. Enel) hanno fatto una scelta *total cloud*: in due anni hanno tolto quasi tutti i loro server e hanno messo tutto su Amazon.

A tutto questo ci siamo arrivati per i tre grandi momenti dell'informatica, ovvero l'informatica dei **mainframe** (anni 50-60-70-80), grandi computer tenuti in modo centralizzato da persone competenti. Dopodiché c'è stato il problema che i mainframe costavano troppo, pochi potevano permetterseli, e grazie all'evoluzione dell'elettronica abbiamo potuto spostare l'informatica del remoto non accessibile al modello **Client-Server**, dove il client è dove ci pare e nel mezzo c'è una rete internet che ci collega a macchine che possono stare ovunque. Questo modello esiste ancora e conveniva avere il server vicino per sfruttare al meglio la connessine. L'informatica distribuita però fa schifo in termini di affidabilità e qualità del servizio. È apparentemente comodo avere il server vicino, ma a chi lo facciamo gestire? Pochi si potevano permettere un ingegnere informatico. Fondamentalmente il primo modello dove c'erano poche macchine ben gestite e presidiate era un modello migliore (quello del mainframe). Nel frattempo, la rete internet è diventata pervasiva e a larga banda. Ci siamo potuti permettere di (se la rete e affidabile e la latenza bassa) ritornare all'informatica seria, quella gestita in datacenter, con la grossa differenza che prima si chiamavano datacenter, mentre oggi ***Internet datacenter***. Infatti, la differenza sostanziale fra il primo e il terzo modello è che il terzo è più flessibile, perché costituito da tanti server, è gestito da persone competenti e collegato alla rete, ottenendo i vantaggi di un sistema distribuito e di un sistema centralizzato. Quindi abbiamo potuto offrire l'informatica come public utility. A noi ingegneri informatici interessa soprattutto il Platform, dove troviamo l'informatica già scritta, dove non saremo sviluppatori, ma integratori. Tutto questo è reso disponibile anche grazie alla potenza di calcolo che è stata creata dagli elettronici e grazie alla rete creata dai telecomunicazionisti. Quindi noi offriamo un IaaS virtualizzato, su cui inseriamo un PaaS.

Il primo che si è inventato questa metodologia è Jeff Bezos. Lui non ha inventato il commercio elettronico, ma voleva diventare il miglior sito di commercio elettronico. La personalizzazione del servizio mancava all'epoca. Ha avuto l'idea che se ho un milione di clienti, devo avere un milione di siti diversi. L'altra cosa che mancava era l'immediato possesso del bene, per cui Amazon è diventata una grande azienda di logistica. L'ultimo aspetto importante è l'idea di dire che se voglio diventare il miglior sito di commercio elettronico, devo investire somme enormi sulle infrastrutture: il sito non deve mai andare giù. Quindi devo investire in infrastrutture informatiche enormi somme di denaro. L'idea che ha avuto è quella di noleggiare le infrastrutture inutilizzate, e poi anche il platform (ad esempio, il sistema di pagamento).

Gli alberghi hanno un costo vuoto per pieno del 70%, ovvero se sono vuoti, costano il 70% dell'albergo pieno. Ci sono, in altre parole, costi fissi. Ecco perché anche gli alberghi offrono molteplici servizi: se ho una lavanderia, perché non posso offrire quel servizio? Se ho una sala conferenza, la affitto, ecc. Oggi un albergo che funziona deve offrire servizi differenziati.

La stessa piattaforma infrastrutturale che regge i siti Amazon di tutto il mondo è quella che offre le macchine virtuali a noi, e in più anche un'offerta platform, ovvero i servizi che lo stesso Amazon utilizza per il commercio elettronico. Essendo stata la prima ha avuto il vantaggio di inventarsi il modello di business per le app: se faccio pagare un'app, prendo una royalties del 30%. Nel momento in cui utilizzi le mie librerie o il mio linguaggio, o te lo fai per te, oppure offri le tue librerie, il tuo software anche sul mercato. Se fai soldi, facciamo 70-30. Essendo stato il primo, tantissimi sviluppatori hanno cominciato a sviluppare per la piattaforma Amazon. Amazon oggi offre un enorme ecosistema di librerie e software fatto da altri sviluppatore. Sicuramente ci si trova qualcosa di utile, sempre. L'ingegnere informatico vede un gran numero di risorse che prima non erano disponibili. C'è elasticità, affidabilità e sicurezza, pagando quello che usi. La sicurezza per i cloud provider è motivo di business. Noi dovremo anche capire come si possono avere queste caratteristiche. Noi vediamo non solo i servizi, ma anche la parte interna.

Il NIST fa standardizzazioni americane. Definisce il cloud computing come un modello per consentire un accesso mediante rete che sia conveniente e on-demand a risorse di calcolo condivise (a livello di rete, server, memorizzazione, applicazioni e servizi) e che può essere fornito e rilasciato in maniera molto rapida (elasticità).

Il cloud provider possiede risorse fisiche, su cui vi è un orchestratore che gestisce i servizi assegnandovi macchine virtuali in modo automatico. Il tutto deve essere configurato. Alla fine, c'è il business, che tiene conto dei consumi e calcola i costi in modo automatico. Trasversalmente c'è la sicurezza e la privacy. Il mondo è più complesso: serviranno nuovi tipi di lavoro, fra cui il lavoro del cloud auditor, che controlla l'effettiva offerta sulla base dei Service Level Agreement, oppure il cloud broker, che è ha una grande esperienza nel panorama cloud e offre ai clienti consulenza. Infatti, c'è troppa distanza culturale fra il consumer e il provider. Quindi serve qualcuno che conosca le esigenze del consumer e che conosca i cloud provider e che in maniera onesta consigli, sulla base delle esigenze, l'approccio da utilizzare.

Cloud providers (1)

I grandi nomi sono Amazon, Google e Microsoft Azure. C'è anche IBM, che creava datacenter, ma all'inizio non si era occupata di cloud providing, ma poi se n'è accorta e si è lanciata, sebbene non avesse a disposizione le librerie. Così come anche Microsoft che è partita in ritardo: Windows e Office sono nell'80% dei PC. Poi ci sono aziende diverse, come Rackspace, che nasce come SaaS, oppure Salesforce, che nasce come CRM, ovvero gestionale per la gestione dei clienti. Da lì è diventato sempre più come un SaaS. Hanno poi rilasciato il loro linguaggio, usando il modello Amazon/Apple. I venditori di connessione, come Verizon e OVH, si sono messi anche loro a offrire ambienti cloud, prevalentemente IaaS. Ci sono anche dei cloud provider specializzati, fra cui Akamai per i servizi di streaming (CDN), e Cloudflare. C'è anche Gartner, che è un'azienda enorme di consulenza e fa i quattro diagrammi, dove tutti vogliono essere in alto a destra. Gli assi dei grafici sono la capacità di realizzare i progetti e la capacità di visione.

Lezione 5

Venerdì 8 marzo 2019

Cloud (3)

Quando sei indietro e vuoi recuperare il passo, o investi o compri altre società. È quello che sta succedendo in tantissimo contesti. Oggi nel contesto digitale tutti erogano gli stessi servizi. Le aziende vogliono evitare di avere molte interfacce, ma vogliono risolvere tutti i problemi. Un altro aspetto interessante è la penetrazione sempre più profonda del cloud nelle varie realtà. Ce ne sono due in particolare, che sono le grandi aziende (enterprise) dove la penetrazione è ormai oltre il 50%, e anche le small-medium business (piccole-medio imprese, ma in termini americani, che sono diversi dall'Italia). L'altro aspetto interessante è quello che abbiamo chiamato community cloud, che avrebbe senso fare nella pubblica amministrazione. La pubblica amministrazione americana spende 80 mld in investimenti informatici. Bisognerebbe capire come spendere meno sulle infrastrutture e più sui servizi: lo facciamo andando verso il cloud. Ha fatto rumore il contratto da 2,5 miliardi di dollari della CIA. Alla fine, sono arrivate in finale Amazon e l'IBM. L'IBM s'era data per vincitrice, ma in realtà ha vinto Amazon. Adesso è in ballo un contratto da 10 miliardi di dollari per il Pentagono. Perché Amazon ha vinto? Per l'ecosistema, per il numero di servizi che Amazon già offriva. Offrendoli da 10 anni, si è sviluppato un numero di librerie e una certa agilità nello sviluppo, test, monitoraggio di applicazioni: è il software il valore aggiunto. Tutta la parte di operations e tutta la parte di servizi che sta intorno sono stati sviluppati per primi da Amazon e quindi si sono perfezionati maggiormente nel tempo. È la stessa cosa che ha consentito a Microsoft di avere successo negli anni 70-80: c'era già tantissimo sviluppato a livello di applicazioni per entrare nel mondo dei PC. È un po' come la Coca-Cola: negli USA, costa meno dell'acqua. Nessuno vende qualcosa simile alla Coca-Cola, e non è colpa degli ingredienti nascosti, ma è il marchio quello che conta e in più non hai la logistica, siccome il costo del prodotto oggi è minimo rispetto alla pubblicità e alla logistica.

Questo vale anche nel mondo dell'informatica: paghi non per il prodotto in sé, ma per tutto quello che c'è intorno. Amazon inoltre a livello di infrastruttura ha sempre investito nel piccolo e smart, a differenza di IBM che ha sempre investito nei mega-computer. Prima Google e poi Amazon hanno investito sul piccolo e sul numero. Stessa cosa sul software: più open possibile e più standard possibile. Hanno investito sull'industrializzazione dell'informatica, col difetto che non è personalizzabile. Questo però ti consente di sviluppare un prodotto che non viene utilizzato solo da te (e quindi costa molto), ma da tante persone, a cui se non piace, se lo fanno piacere perché costa meno. Riassumendo: Amazon è ancora considerato il migliore perché facilita l'uso rapido dello sviluppo tramite il più ricco ecosistema di applicazioni e librerie e in tutti i linguaggi possibili, quindi riesce fare i prezzi più bassi, perché cerca la standardizzazione per avere prezzi più bassi, sia a livello hardware, che rete, che software.

L'**hype-cycle** è il destino delle tecnologie: prima seguono una curva di crescita, che alcune tecnologie non oltrepassano mai. Quando arrivi in cima, c'è la curva della disillusione, e poi c'è una leggera crescita della tecnologia e non si parla più di ricerca in quell'ambito, ma di un prodotto già sul mercato. Abbiamo parlato di IaaS, PaaS e SaaS. Oggi è più complicato: perché venderti solo dell'hardware? Per lo meno ci devi mettere un sistema operativo. Non è IaaS, non è proprio PaaS. Anche aziende che sono nate come SaaS, su esigenza dei clienti hanno introdotto la possibilità di mettere a disposizione il proprio linguaggio (come Salesforce, che ha messo a disposizione dei clienti il proprio linguaggio M-Force). Non sono quindi un PaaS vero e proprio. Quindi sia SaaS che IaaS forniscono servizi aggiuntivi, avvicinandosi a PaaS. C'è chi lo chiama IaaS+. Oppure un IaaS che fornisce servizi specifici per alcune realtà. I provider sono molto contenti di fornire servizi arricchiti. **Sceglierei IaaS se** volessi dare importanza alla availability, alla scalabilità e ai costi. L'availability è collegata ai costi, perché a seconda delle esigenze del servizio (deve funzionare sempre, solo in alcune fasce, ecc.) il costo varia. Se il sistema deve funzionare h24, devo mette sul conto ad esempio 4 stipendi uomo per la gestione dell'infrastruttura. Oggi ovviamente nessuno si accontenta di fornire un servizio non disponibile tutto il giorno. **Sceglierei PaaS se** volessi dare importanza ad aggiornamenti e patch management, implementazione di nuove applicazioni, availability, scalabilità, costi. Sui costi è un po' meno chiaro: senza dubbio sui costi di sviluppo. Invece usabilità e sicurezza dipendono da chi implementa, quindi non posso contarci. **Sceglierei SaaS se** volessi dare importanza alla sicurezza e usabilità, perché chi fa software su queste piattaforme parte dal principio che non deve esistere il manuale utente. Nel momento in cui noi usiamo questi servizi, paghiamo per quanto i clienti usano (non noi). Quindi più clienti abbiamo, più il contatore aumenta. Perciò se non vogliamo fallire, più utenti abbiamo, più dobbiamo guadagnarci. Quindi devo innanzitutto analizzare il mio modello di business, perché se non posso guadagnarci, è meglio non pagare per ogni cliente che utilizza il mio servizio.

Cloudification (1)

Il cloud si sta mangiando tutto il resto. È stato previsto che entro il 2021 il 94% del carico di lavoro sarà in un qualche ambiente cloud. Stiamo producendo tantissimi dati e dobbiamo preoccuparci del dove andiamo a metterli. I dati poi devono essere elaborati per poter ricavare informazioni. Quindi serve potenza di calcolo elastica e disponibile. Se fai un servizio, speri di avere successo. Se fai un servizio per i tuoi colleghi, vuoi che il tuo servizio sia sempre disponibile in tutto il mondo per tutti i dipendenti che vuoi.

Lezione 6

Lunedì 11 marzo 2019

Cloudification (2)

Il mondo e il business richiedono sempre più flessibilità: l'informatica deve allinearsi a questo cambiamento e non deve frenare l'innovazione. Quindi sempre più dovremo cercare di integrare questo mondo fatto di città, industria e oggetti smart, strumenti che in qualche modo saranno dotati di capacità di micro-elaborazione e connessione. La visione iniziale era quella che tutto finirà nel cloud. Tutto questo in un mondo dove dovremo gestire i fornitori, i cui prodotti potranno essere tenuti sotto controllo minuto per minuto dal cliente. Potremo e dovremo tamponare eventuali problemi quindi. Se abbiamo una produzione sempre più smart e legata alla rapidità di prodotto, potremo tamponare eventuali anomalie. Dobbiamo conoscere materie come AI, mobile workforce, sensori, cyberphisical systems, cloud, IoT, cybersecurity, ecc. Quindi sistemi, fabbriche e manifatture che si adattano dinamicamente alle necessità del cliente, senza mai abbandonare il cliente. Il confine di queste materie non c'è, perché sono tutte collegate fra loro e tutto serve. Anche il cloud evolve, quindi è sempre più utile offrire **Integration PaaS (iPaaS)** per l'integrazione. Oggi si cercano quindi persone che sappiano integrare qualsiasi cosa. È importante capire qual è il flusso di dati, quali sono i processi. Quindi bisogna capire da dove arrivano i dati e quale sia il valore aggiunto che possiamo dare ai dati, e solo dopo pensare agli strumenti. Gli **Application PaaS (aPaaS)** danno qualcosa in più rispetto alla mobilità, con il vantaggio del minor tempo. Col cloud il software open-source ha trovato un secondo motivo di esistenza, perché se posso lo uso open lo faccio non tanto per modificarlo a mio piacimento, ma per non pagare licenze (soprattutto licenze al consumo). L'altro aspetto è che non elimini, ma senza dubbio riduci il vendor lock-in. Ormai tutto è "aaS". Anche la sicurezza. Oggi va di moda il backup as a service: il backup si fa sul cloud. Altri spostano sul cloud la marea di dati prodotti da strumenti di controllo e monitoraggio.

Mobilità

Il mondo sta diventando tutto mobile. C'è un crollo sui desktop generale e il mobile, sia come smartphone che come tablet, sia come laptop, ha superato i dispositivi fissi. Questo ha delle conseguenze per chi deve sviluppare servizi: non solo devi fornire le nostre caratteristiche, ma le devi fornire soprattutto a utenti che si spostano in modo differente. La maggioranza delle persone ha bisogno di spostarsi e usare servizi mentre si muove. La maggior parte del traffico sarà mobile e di tipo immagine/video. Quindi vogliamo un'informatica *anywhere, anytime, anydevice*. I nostri servizi informatici dovranno essere disponibili sempre, dovunque e da molteplici dispositivi. Noi ci portiamo sempre i dati con noi: chiavette, dischi, laptop. Non è una scelta sana nel futuro, perché i dati possono essere persi. Sui dispositivi mobili ci sono sia informazioni personali che informazioni aziendali, che spesso non sono divisibili nettamente. La domanda è: il cloud può essere un'alternativa? L'informazione è denaro, mentre i dati sono costosi, per acquisirli, immagazzinarli, ecc. Se l'informazione è denaro e non proteggi l'informazione, allora stai perdendo denaro. I cloud sono le banche del presente e del futuro. L'importante è reperire i propri dati quando ce n'è bisogno, analogamente ai soldi. In un ipotetico futuro, i laptop potrebbero non avere disco, così come le carte di credito non hanno i soldi. Il disco del cloud entra in tutto e per tutto a far parte della gerarchia dei dischi di un calcolatore.

IoT: fog computing (1)

L'evoluzione nella connettività ha fatto sì che l'informatica sia pervasa nella vita dei tutti i giorni. Dall'assistente vocale, alla guida autonoma, ai sensori per la casa, ai sensori per l'industria, ecc. Anche Amazon ha creato Libelium per i sensori IoT. Il problema è che questi milioni di dati forniscono dati, abbiamo dati che vengono presi e ritornati. Dove vanno a finire? Il cloud va bene per tutto? Nelle applicazioni real-time, la latenza, la disponibilità e l'affidabilità della rete non possono soddisfare i requisiti di queste applicazioni. Quindi nasce il concetto di **fog**: è una nuvola più vicina alla terra. Quindi le applicazioni che sono latency sensitive hanno la necessità di fornire servizi di memorizzazione e processamento all'*edge*, vicino a dove questi dati vengono gestiti.

Lezione 7

Martedì 12 marzo 2019

Cloudification (3)

IoT: fog computing (2)

Ieri abbiamo concluso dicendo che il cloud non è la soluzione ottima per tutto. Infatti, uno dei problemi del cloud è l'affidabilità e la latenza della rete: ci sono applicazioni che non sono molto sensibili alla latenza e ce ne sono altre che lo sono. Si è parlato quindi di **fog computing**: avere dei server più vicini a dove i dati vengono generati e dove si attendono risposte. Da cui il termine **edge**. Sono la stessa cosa? Qual è la **differenza fra fog ed edge**? Non si sa, ognuno dà la propria interpretazione. Sono sinonimi, oppure uno è sopra all'altro, oppure gli edge server sono quelli che realizzano il fog, oppure il fog va bene per l'IoT, mentre l'edge server va bene per l'industrial IoT. Se devi vendere un server industriale è più facile vendere un edge server. Tipicamente a livello di server si parla sempre più spesso di edge server. Il tutto nasce come fog, oggi sono un po' in alternativa. Quando uno progetta/realizza un'architettura fog può crearne una indipendente dalle applicazioni, altrimenti si possono creare architetture specifiche per le applicazioni. Le fog sono promosse molto anche nel settore pubblico, per ciò che è (genericamente) smart city (concetto interpretato diversamente da ogni città), tipicamente legato alla gestione del traffico, alla qualità dell'aria, al riciclo dell'immondizia, ecc. Il concetto è che noi abbiamo almeno tre livelli: fra i device e il cloud si aggiunge un livello intermedio di nodi fog, che in realtà sono molto più vicini ai device. Un'altra architettura possibile è simile a quella precedente, ma verticalmente si aggiungono nodi edge per le applicazioni industriali. Non è detto che poi questi nodi vadano sul cloud. L'idea è proprio sul discorso rete: aumentare l'affidabilità e riducendo la latenza. Un altro aspetto importante riguarda i dati. Se lo storage è nel mio server, i dati sono privati, mentre per i dati sul cloud non possiamo sapere.

Sta succedendo quello che è spesso successo nell'informatica: dal mainframe, al Client-Server, dal cloud, all'edge computing. Si è passato dal modello centralizzato a quello decentralizzato ben due volte. Due considerazioni: il nuovo modello e le nuove tecnologie non rimpiazzano quelle precedenti (esistono ancora mainframe nelle banche, nei comuni e in qualche azienda); non è vero che sono mode. Perché si segue un andamento sinusoidale? Cos'è cambiato? La rete. Nel passaggio fra mainframe e Client-Server è nato internet, ma era molto scarso ed inaffidabile. Il passaggio al cloud computing è avvenuto per la WAN che è diventata pervasiva, affidabile e con latenza bassa: il client non si rende conto della distanza dal server. Ma per alcune applicazioni la rete deve sempre funzionare e con bassa varianza: parliamo di applicazioni real-time. Quindi si è passato all'edge computing. La situazione attuale è questa: i client sono smartphone, tablet, laptop, desktop; i server sono schede (Arduino, Raspberry), rack, edge server, data center. Con **Operation Technology (OT)** si intende un sistema con massima reliability, protegge il funzionamento. Si distingue quindi da **Information Technology (IT)**. Quali sono le priorità di una o dell'altra? Per IT sono scalability, data integrity,