Università degli Studi dell'Aquila
Facoltà di Ingegneria

Tesi di Laurea Specialistica in Ingegneria Informatica e Automatica

Realizzazione di un prototipo della versione Cloud SaaS della suite IBM BigFix: Automazione del deployment e del testing

Relatore interno:
Prof. Serafino Cicerone

Laureando:
Beniamino Negrini

Correlatore:
Prof. Dott. Marco Secchi

Relatore esterno:
Dott. Bernardo Pastorelli

Anno Accademico 2016/2017

Dedica a piè pagina

Indice

Capitolo 1

Introduzione

(prima bozza)

E' sempre più evidente che il cloud computing è il futuro del software. La rivoluzione consta nella distribuzione dei servizi di calcolo e nella virtualizzazione delle risorse, dando così all'utente la sensazione di un utilizzo centralizzato. Tutto ciò si è reso realizzabile dal momento in cui l'accesso alla rete è divenuto possibile da sempre più dispositivi e con velocità di connesione sempre maggiore.

la tematica del cloud computing è stata centrale nel mio lavoro di tesi presso l'azienda IBM (International Business Machines Corporation) nella sua sede di Roma. Ho partecipato attivamente alla realizzazione di un prototipo software, ossia la versione SaaS della suite IBM BigFix. BigFix è una suite di prodotti dedicati alle aziende che risolvono problematiche di Endpoint Security e di compliance di dispositivi interni a determinate politiche aziendali. Tramite questi prodotti si ottiene pieno controllo su tutti i dispositivi aziendali, si rilevano eventuali attacchi, si distribuiscono aggiornamenti e patch fino anche alla gestione energetica.

La sfida da me raccolta è quindi proprio quella di portare tutto questo arsenale di strumenti nella leggerezza del cloud. Rendendolo disponibile, nel giro di pochi minuti, anche a chi è sempre stato intimorito dalla difficoltà di installazione di uno strumento così potente, ma allo stesso tempo complesso.

Capitolo 2

IBM BigFix

2.1 BigFix

I prodotti della suite IBM BigFix consentono di monitorare e gestire in tempo reale un elevato numero di dispositivi fisici e virtuali connessi (fino a 250.000). Questi possono essere sia fisici che virtuali, come ad esempio server, desktop, notebook, dispositivi mobili, tablet, POS, ATM e chioschi self-service. Gli utenti principali di questi prodotti sono gli amministratori di sistema. Tramite le applicazioni BigFix possono avere il pieno controllo sugli endpoint. Possono ad esempio distribuire software, applicare delle patch, effettuare il deploy di sistemi operativi, proteggere da attacchi di rete e molto altro.

2.1.1 Architettura

L'architettura di BigFix è, per sua natura, molto articolata, poichè la necessità è quella di gestire un numero elevato ed eterogeneo di dispositivi. Essa si basa sul consolidato pattern stilistico Client/Server, ma con una struttura leggermente variata, prevedendo l'inserimento di un ulteriore layer frapposto tra client e server, i relay, i quali sono fondamentali per bilanciare il carico.

Ma partiamo subito con un esempio per avere un ponto di riferimento. Come possiamo notare, l'elemento fondamentale è il server, il quale ha lo scopo di racco-

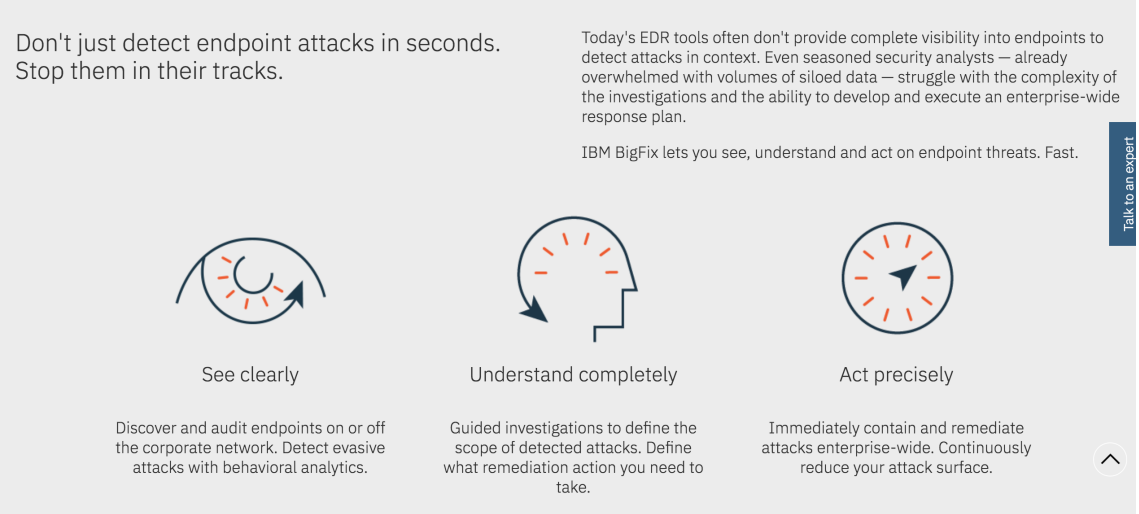

Figura 2.1: Breve header descrittivo di BigFix dal sito web di IBM Security

gliere dei particolari messaggi, le Fixlet. Questi messaggi posso essere visualizzati dall'operatore che lavora sulla console e inoltrati poi, a questo punto ai relay. E' competenza dei relay, a questo punto, interagire con i singoli client e assicurarsi l'esecuzione delle Fixlet. Le Fixlet, infatti, altro non sono che delle azioni che devono essere necessariamente compiute dai client. Andiamo ad analizzare le singole componenti dell'archiettura.

Servers Il server coordina tutto il flusso di informazioni e si preoccupa di salvare le informazioni sul database. Al tempo stesso però, lascia agli Agent il compito di effettuare analisi ed eseguire azioni specifiche. Ciò consente di liberare il Server da un pesantissima computazione. Per questo motivo il Server stesso può gestire un altissimo numero di client.

Relays I Relay si comportano come una cache tra i Client e il Server e sono di numero variabile in base al numero di Client. Aiutano il Server a gestire i dispositivi anche se funzionalmente non sono altro che Client che sono stati promossi a Relay, aggiungendo a loro dei servizi. A questo punto i Client non si interfacceranno mai con il Server, alleggerendone così notevolmente il workload. Possono, ad esempio,

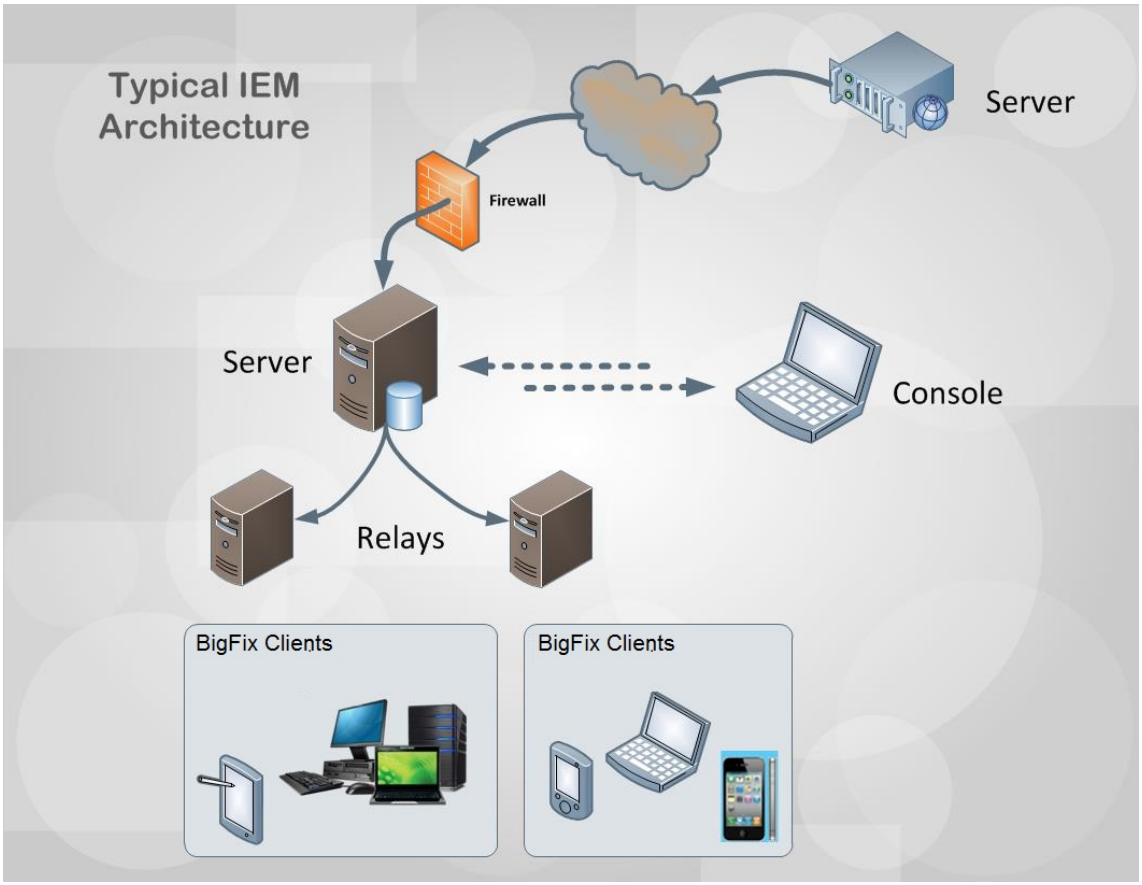

Figura 2.2: Un'architettura BigFix di esempio

più Client richiedere un download al Relay, il quale effettuerà un'unica richiesta al Server

Agents Un Agent è installato su ogni Client facente parte dell'architettura di BigFix. essi hanno il compito di raccogliere le Fixlet, tramite le quali sono in grado di compiere tutte le azioni necessarie. Un Agent fa dei continui check per confrontare lo stato del dispositivo con le policy stabilite. Appena scopre che il dispositivo è fuori dalla compliance, viene informato il Server ed agisce subito per porre rimedio e, al termine dell'attività, l'Agent informa nuovamente il Server sull'esito dell'operazione.

Web Reports I Web Reports costituiscono il componente che consente ad utenti autorizzati di monitorare tutti i dispositivi di BigFix. Si può, in questo modo, tenere

traccia di vulnerabilità, azioni richieste e molto altro.

Consoles La Console permette agli amministratori di interagire con tutti i Client dell'ambiente BigFix. Gli utenti possono così distribuire velocemente patch e configurazioni.

BigFix, da un punto di vista logico, si suddivide in due grandi macro-componenti, la Platform e le Applications. La prima svolge la funzione di layer sopra la quale vengono sviluppate tutte le funzionalità dello strato di applications. Questa suddivisione consente una chiara suddivisione delle competenze da parte di progettisti, sviluppatori, tester e assistenti dei clienti. Il team della Platform si concentra quindi nel fornire una solida infrastruttura al team delle Applications, il quale svilupperà i singoli strumenti al servizio dell'utente.

2.1.2 BigFix Platform

La Platform è una tecnologia multi-layer scritta in linguaggio C++ che agisce come colonna portante di tutta l'infrastruttura di BigFix. Essa svolge infatti funzioni fondamentali, spesso utilizzate anche da altre applicazioni dei layer superiori. Le attività della Platform si dividono in base alle differenti responsabilità.

Lifecycle Management Include il controllo remoto, la distribuzione dei softawre e il deploy di sistemi operativi.

Patch Management Consente l'applicazione e la gestione delle patch.

Core Protection E' il cuore di tutta la routine di Security. Troviamo quì delle funzionalità anti-malware, firewall e protezione dai virus.

Inventory Raccoglie informazioni peculiari a riguardo dei software installati su ogni dispositivo. E' in grado di effettuare analisi sugli utilizzi fornendo una base per la decisione sulle licenze da acquistare.

Server Automation Fornisce un Hypervisor per le Virtual Machines controllando, infatti, infatti eventuali malfunzionamenti degli ambienti virtuali.

2.1.3 BigFix Applications

Tutti i prodotti applicativi che fanno parte di questo componente consentono di gestire in maniera semplice le operazioni inerenti alla security. A differenza della Platform, sono implementate in linguaggio Java ed hanno funzionalità atomiche tra di loro. Sono l'interfaccia principale con il quale interagisce l'amministratore aziendale.

BigFix Lifecycle Questa è l'applicazione che l'amministratore utilizza per gestire il ciclo di vita degli endpoint fisici. Ha una visibilità completa su di essi e pone rimedi immediati. Tra le funzioni principali ci sono quelle di power management, software distribution e OS deployment.

BigFix Detect Detect consente di effettuare analisi approfondite sulla security degli Endpoint. Comprende meccanismi avanzati per individuare in real-time attacchi agli Endpoint investigando attività sospette e violazioni di sicurezza. Fornisce infine degli strumenti per porre un repentino rimedio a questi incidenti.

BigFix Patch E' l'applicazione che consente la distribuzione di patch sia a livello di applicativi che a livello di sistema operativo.

BigFix Compliance Si utilizza questa applicazione per garantire la compliance dei dispositivi, identificare irregolarità e risolverle.

BigFix Protection Questa applicazione viene adoperata per garantire una protezione real-time contro qualsiasi genere di malware (virus, trojan, worms, spyware, rootkits e altre minacce web). Ha ovviamente effetto sia sugli endpoint fisici che sulle maccchine virtuali.

BigFix Inventory Come dice la parola stessa, questa applicazione si occupa di analizzare gli endpoint e generare un inventario di tutti i software su essi installati, generando dei report ed evidenziando, eventualmente, le irregolarità.

2.1.4 Fixlets

Le Fixlet sono il metodo attraverso il quale si svolgono tutte le operazioni, come distribuzione di software, installazioni di patch e configurazioni. Esse sono dei messaggi inoltrati ai Client di BigFix e utilizzano un linguaggio di query specifico, il Relevance.

Il linguaggio Relevance

Con una Fixlet si può anche ispezionare un desiderato aspetto di un Client. A tale scopo viene adoperato il linguaggio Relevance. Esso, infatti, consente di interrogare il Client, identificandone caratteristiche dell'hardware o del software tramite particolari costrutti, gli Inspectors. Una necessità può essere infatti quella di applicare una Fixlet solamente a dei client con determinate caratteristiche hardware/software oppure che si trovano in stati ben definiti. Si può, in questo modo, facilmente identificare il corretto sottoinsieme di Client ai quali è destinata una nuova Fixlet ed applicarla solo ad essi.

2.2 IBM

Come detto, il lavoro di tesi si è svolto nell'ambito di un progetto formativo stipulato tra l'Università dell'Aquila e IBM Italia Spa. Questo progetto ha previsto un tirocinio svolto nella seda di Roma con obiettivo: "Esplorazione e prototipazione di metodi per portare prodotti BigFix su cloud", per l'appunto la realizzazione del prototipo di BigFix Saas.

La storia dell'IBM ha inizio nei primi decenni del novecento, ma è dagli anni settanta che entra nel mercato dell'informatica, soprattutto nel settore hardware. Negli ultimi venti anni il business si è spostato sempre più sul software. In particolare soluzioni cognitive e piattaforme cloud.

IBM Security Presso la sede di Roma presente il più importante laboratorio IBM italiano, il Rome Softare Lab. Nella divisione italiana ci si concentra prevlentemente sullo sviluppo back-end. Una grossa fetta del laboratorio si occupa della divisione Security di IBM. Il portfolio di Security contiene per l'appunto prodotti che si occupano di diversi aspetti della security aziendale, tra questi BigFix è uno dei pù consolidati.

2.3 Saas Exploration Project

Lo scopo del progetto al quale ho partecipato con il mio lavoro è quello di esplorare le tecnologie esistenti nel panorama cloud e realizzare il prototipo della versione SaaS di BigFix. A questo scopo, oltre a me, sono state allocate altre tre persone full time al conseguimento del progetto, sotto la guida dell'architect Bernardo Pastorelli.

2.3.1 Il framework SCRUM

Il team adotta il framework agile SCRUM. Questo modo di operare è di sempre maggiore diffusione ed è basato su un approccio iterativo e incrementale nello sviluppo software. Il design e lo sviluppo sono divise in iterazioni, denominate "Sprint", della durata fissa di due settimane. Queste due settimane terminano sempre con una versione funzionante del prodotto, il quale viene mostrato in una demo che ne evidenzia le nuove features implementate.

SCRUM utilizza un approccio empirico alla progettazione. La filosofia di fondo del framework è quella che la conoscenza deriva dall'esperienza, e quindi tutte le scelte che si prendono nel corso della progettazione devono avvenire alla luce di una sempre maggiore esperienza, la quale si ottiene avendo a disposizione il prima possibile un sottoinsieme del prodotto a se stante testabile ed usabile. Dì quì l'approccio fortemente iterativo e incrementale massimizzando le opportunità di feedback.

All'inizio del progetto vengono definiti i requisiti del prodotto (item), i quali vengono da un'attenta analisi dei bisogni dell'utente. Ogni bisogno viene modellato con una Epica, che a sua volta viene prioritizzata e aggiunta al Product Backlog che le

indicizza. Le Epiche vengono poi scomposte in User Stories, le quali si suddividono a loro volta in Task, ossia l'elemento atomico del progetto la cui implementazione viene presa in carico da un singolo componente del team.

L'inizio di uno Sprint è sempre caratterizzato da un meeting in cui si pianificano gli obiettivi per il prossimo Sprint. In questo contesto si fa sempre riferimento al Backlog incentrandosi sulle User Stories ancora non coperte. Si cerca quindi di suddividersi i Task in modo tale da avere a fine Sprint quelle nuove funzionalità usabili e dimostrabili. Demo che viene svolta sempre con la presenza di tutto il team e anche di colleghi di altri laboratori IBM.

2.3.2 Sistemi di controllo di versione

Da un'organizzazione di questo tipo ne scaturisce la necessità di tool di controllo di versione che permettano una fluida gestione del codice e della programmazione in parallelo tra i diversi componenti del team.

GitLab A tal scopo si è adottato ormai da tempo, da tutto il team di BigFix, il software di controllo di gestione distribuito Git e una repository aziendale che consiste in una versione enterprise ad hoc per IBM di GitLab, un hosting service molto simile a GitHub.

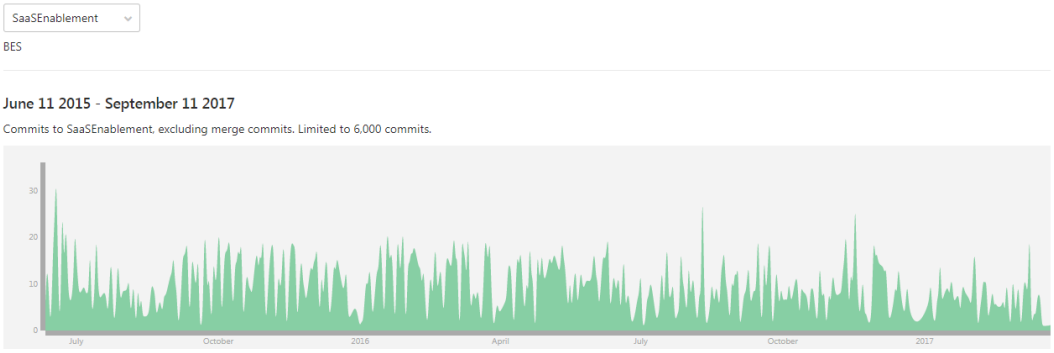

Figura 2.3: Panoramica degli ultimi contributi nel branch del progetto SaaS

Il flusso di lavoro è il seguente. Quando inizia il proprio task, il componente del team si pone su un proprio branch personale sul quale effettua i propri commit. Al termine del task viene fatta una merge request sul branch principale, sono una volta che si è testato il codice, per aggiungere i propri contributi al progetto. A questo punto, dopo una review effettuata da componenti del team accreditati, il nuovo branch verrà mergiato con il branch principale.

2.3.3 RTC

E' ovviamente necessario un tool che coordini anche la suddivisione dei task all'interno del team. A tal fine abbiamo utilizzato un software interno di IBM, ovvero Rational Team Concert (RTC). Esso offre comodi strumenti di agile planning e gestione di ciclo di vita del software. Ogni componente può così tracciare facilmente le aree di sua competenza. E' possibile inoltre usufruire di tool per il source control, controlo dei difetti e gestione delle build.

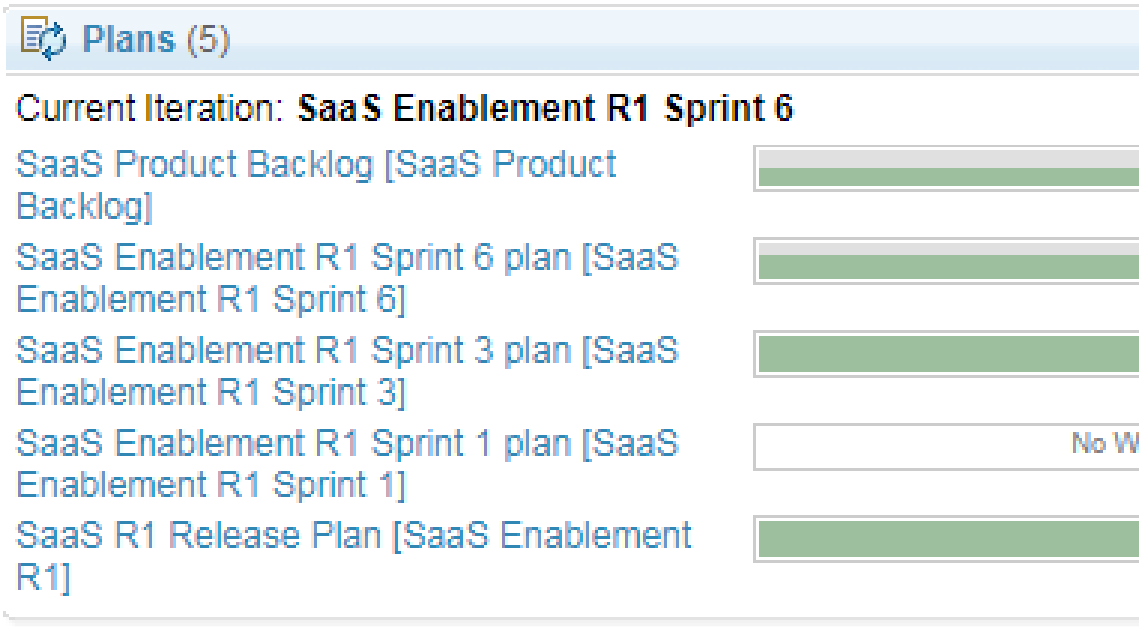

Figura 2.4: RTC. Un'esempio di come viene monitorato il completamneto dei diversi sprint

Capitolo 3

Il Cloud e le sue sfaccettature

3.1 Cloud Computing

La differenza tra il possedere e l'utilizzare. E' questo l'aspetto cruciale del cambiamento apportato dal cloud computing rispetto al software tradizionale. Le risorse, che siano esse stesse archiviazione, elaborazione o qualsivoglia risorsa informatica, non sono mai ad hoc per un singolo utente, ma vengono assegnate on demand a diversi utenti e appartengono ad un insieme condiviso da tutti gli altri utenti del prodotto utenti. Attraverso internet ogni utente può accedere a queste risorse in qualsiasi momento. Tali risorse vengono opportunamente allocate all'utente in maniera dinamica e completamente automatizzata. L'utente può utilizzare così anche software non installati sul proprio computer o usufruire di una memoria di massa accessibile da parte sua da qualsiasi dispositivo.

L'esperienza utente che si vuole fornire però è quella di un utilizzo esclusivo della risorsa, come nei software tradizionali, mentre in realtà la risorsa viene solo sapientemente distribuita tra gli utenti. Ciò fa si che, potenzialmente, un singolo utente possa acquisire risorse notevolmente maggiori nel caso medio.

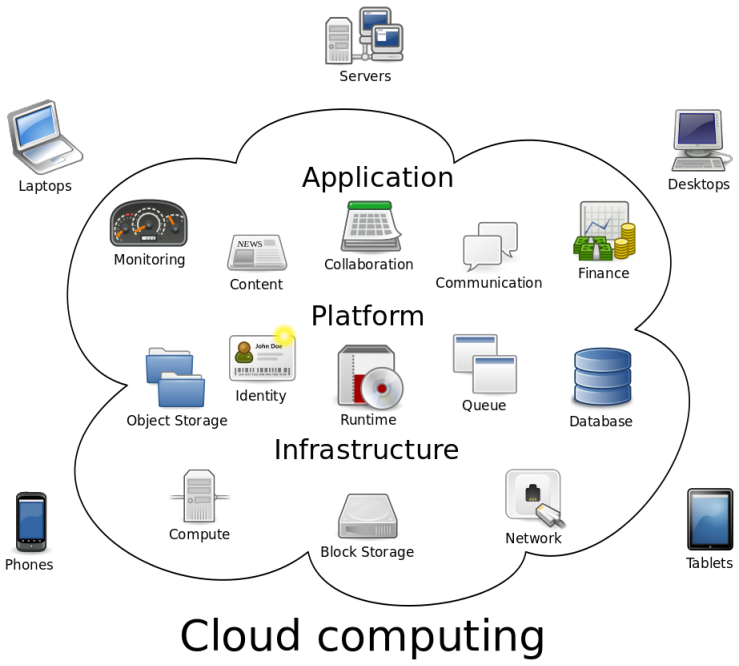

Figura 3.1: Diagramma logico di una rete Cloud Computing

3.1.1 Vantaggi del Cloud Computing

- Costo:
  Con l'avvento del cloud tutta la gestione dell'infrastruttura sottostante al software diviene a carico del provider. Vengono eliminate spese per la gestione dei data center locali. Facendo riferimento alla versione SaaS di BigFix ad esempio, il cliente viene sollevato dal pesante onere di utilizzare server locali e gestirne le relative connessioni. Il provider detiene tutto l'hardware di cui il cliente ha bisogno.

- Velocità:
  Anche quì ci risulta molto utile prendere come esempio la suite di BigFix. quando un nuovo cliente acquista il prodotto nella sua versione on-premises, un'incaricato di IBM si reca presso il cliente e lì inizia un lungo processo di installazione della suite che può impiegare diverse ore. Nello scenario SaaS il cambiamento è radicale. E' sufficente che il cliente compili una form online, dopo alcuni minuti poi riceve una mail con il link per accedere al servizio.

- Prestazioni
  Una delle motivazioni principali per la quale si sceglie di fare uso del clud computing sono proprio le prestazioni, soprattutto se si adotta il paradigma PaaS. Esternalizzando le risorse di calcolo, si può fare affidamento a dei provider che fanno dei server ad alte prestazioni il loro punto di forza. L'utente può, in questo modo, abbattere dei bottleneck che altrimenti risulterebbero di grande impedimento. Nel 2016 IBM mette per la prima volta a disposizione pubblicamente un computer quantico, proprio attraverso una piattaforma cloud (IBM Q). Questo può rappresentare un esempio estremo in ottica prestazioni, ma che può rendere un'idea di quale potrà essere il trend nei prossimi anni.

- Affidabilità
  Operazioni di mirroring da parte dei provider dei servizi cloud fa sì che il backup dei dati sia continuo ed economico.

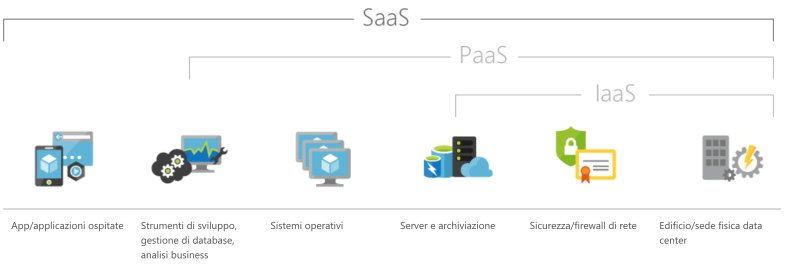

Figura 3.2: Panoramica delle principali tipologie Cloud

3.2 Tipologie di servizi Cloud

Il termine Cloud risulta in realtà molto generico. Esso comprende diverse tipologie di fornitura dei servizi, a seconda della risorsa che viene offerta dal provider. La maggior parte dei servizi di Cloud Computing rientrano in tre tipologie principali: Infrastruttura distribuita come Servizio (IaaS, Infrastructure as a Service), Piattaforma distribuita come Servizio (PaaS, Platform as a Service) e Software come un Servizio (SaaS, Software as a Service). Oltre a queste tipologie, annoveriamo anche soluzioni minori come il DaaS (Data a a Service) e l'HaaS (Hardware as a Service). Andiamo a vedere nel dettaglio come, a seconda della modalità di utilizzo del paradigma Cloud, questi paradigmi si differenziano.

3.2.1 IaaS, Infrastructure as a Service

E' la tipologia più basilare. Vengono messe a disposizione piattaforme di elaborazione. Utilizzando un IaaS si affittano le infrastrutture utili ai propri scopi, come add esempio Server, macchine virtuali (VMs), risorse di archiviazione, reti e sistemi operativi. Può, inoltre, essere messo a disposizione anche hardware in remoto. Il provider di servizi cloud gestisce l'infrastruttura, mentre l'utente acquista, installa, configura e gestisce il software, tra cui sistemi operativi, middleware e applicazioni.

Vantaggi Una soluzione IaaS è quella che garantisce maggiore flessibiltà. Tra i vantaggi principali ricordiamo:

- Elevata scalabilità
  Il modello IaaS permette una scalabilità verticale rapida ed economica

- Rapidità di innovazione
  Nel caso del lancio di un nuovo prodotto basato sulla piattafroma IaaS, il tempo di attesa per le nuove configurazioni infrastrutturali è solamente dell'ordine di pochi minuti.

- Adattabilità alle richieste
  Un modello IaaS è estremamente flessibile alle variazioni delle richieste. Si possono facilemnte aumentare le risorse nei momenti di picco e ridurle quando non è necessario, risparmiando quindi denaro.

3.2.2 PaaS, Platform as a Service

Una piattaforma distribuita come servizio (PaaS, Platform as a Service) è un ambiente cloud di sviluppo completo. Una soluzione PaaS è progettata per consentire il ciclo completo dello sviluppo delle applicazioni: creazione, test, distribuzione, gestione e aggiornamento. L'utente ha tutta la libertà di sviluppare gli applicativi a proprio piacimento, ma lavora con componenti software già pronti all'utilizzo (Microservices). Questi componenti non sono localizzati presso chi utilizza il cloud, bensì presso il provider, il quale si occupa del loro mantenimento e aggiornamento. Il modello PaaS consente di evitare le spese e la complessità legate all'acquisto e alla gestione di licenze software, middleware e infrastruttura delle applicazioni sottostanti o strumenti di sviluppo. L'utente gestisce le applicazioni e i servizi che sviluppia e il provider cloud si occupa di tutto il resto.

Vantaggi Uno scenario PaaS riduce quindi notevolmente la quantità di codice da scrivere semplificando quindi il lavoro dello sviluppatore e aumentandone la produttività. Inoltre risulta molto più semplice così il porting di un prodotto da web a mobile e viceversa. Componenti molto complessi e costosi possono inoltre essere messi a disposizione, con un utilizzo limitato, anche per sviluppatori che altrimenti non potrebbero permetterselo.

IBM Bluemix

Troviamo, sempre all'interno di IBM, uno dei principali servizi cloud PaaS presenti sul mercato: IBM BlueMix. L'utente può usufruire di un'astrazione di molte componenti utili allo sviluppo. Si può, ad esempio, fare uso di Database specifici, di moduli dedicati all'IoT, di tecnologie Blockchain e molto altro. Ne vediamo alcuni esempi nella figura 3.3

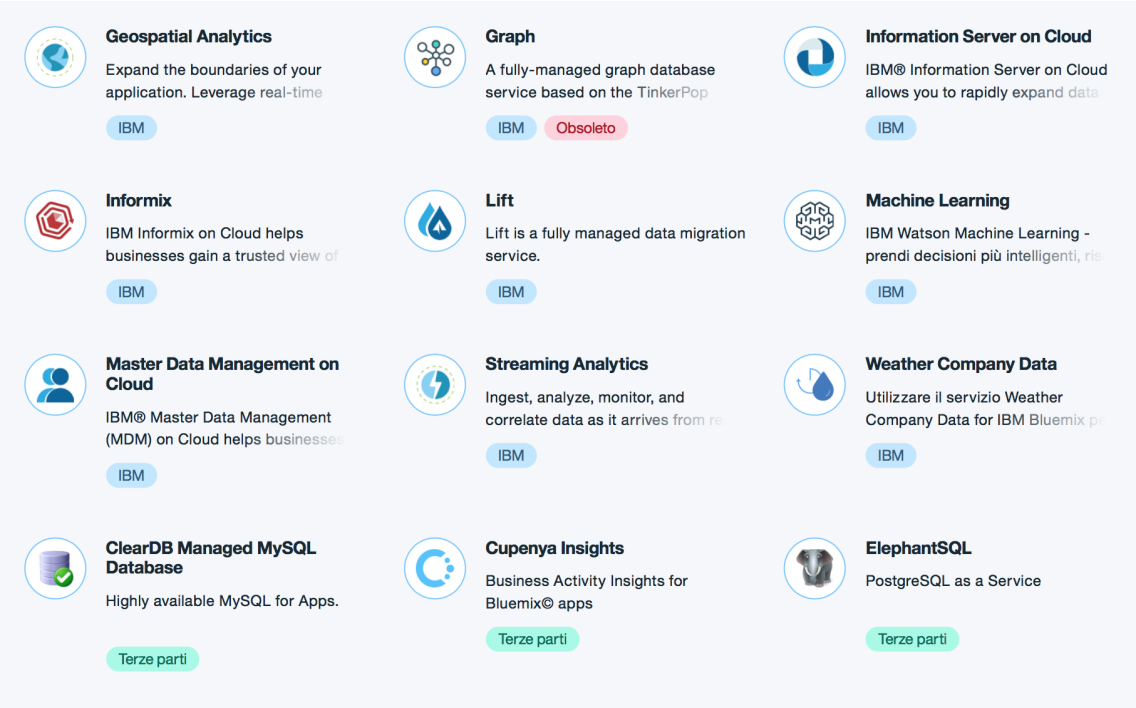

Figura 3.3: Esempi di moduli presenti nel catalogo BlueMix

IBM Watson

Tra le componenti sviluppate da IBM merita una menzione anche Watson. Watson è un sistema di intelligenza artificiale in grado di rispondere a domande espresse in un linguaggio naturale. Tra le funzionalità ci sono quelle di elaborazione del linguaggio naturale, information retrieval, rappresentazione della conoscenza, ragionamento automatico e tecnologie di apprendimento automatico.

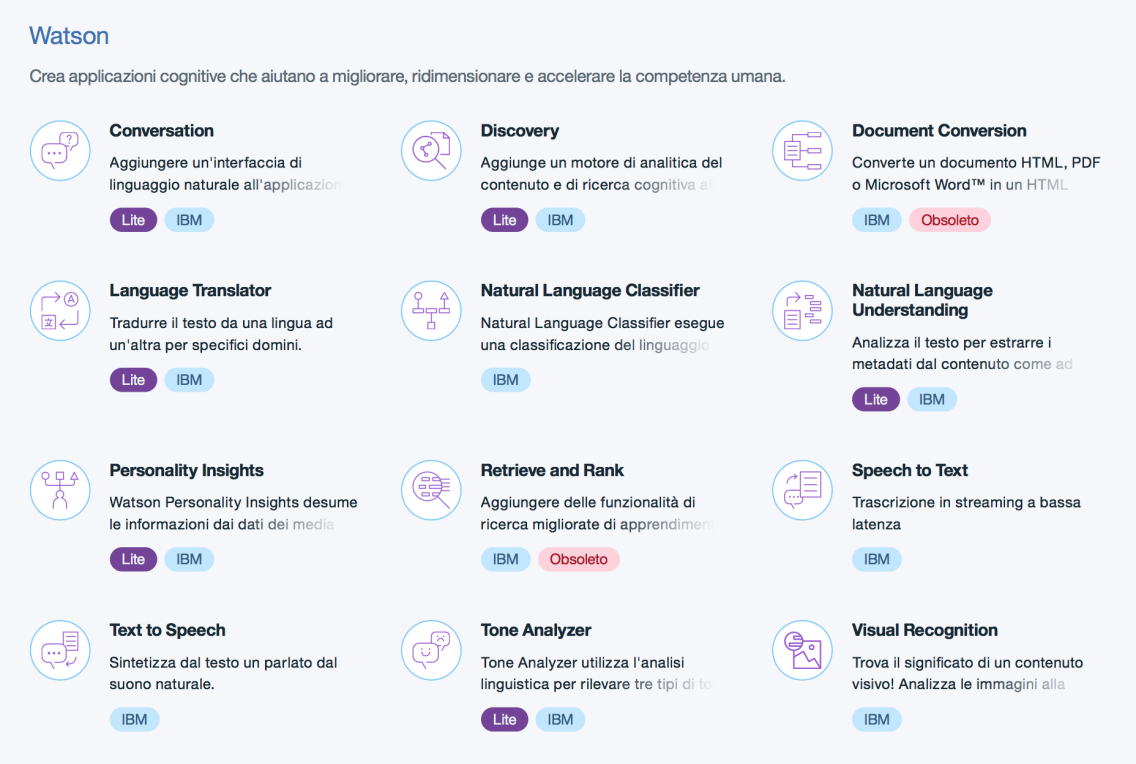

Figura 3.4: Alcuni moduli Bluemix appartenenti a Watson

3.2.3 SaaS, Software as a Service

Il Software as a Service (SaaS) è un modello di distribuzione del software in cui l'applicativo e gli eventuali servizi collegati sono eseguiti in un ambiente centralizzato e gli utenti vi accedono via rete, quasi sempre via Internet e usando un browser come interfaccia. I SaaS sono ormai sempre più diffusi. Tra i maggiori si ricordano le Google Apps (ad esempio Gmail, Google Drive, Google Calendar) e la suite Microsoft Office 365. Con la metodologia SaaS il provider fornisce tutto il software direttamente all'utente, al quale non resta che usarlo senza preoccuparsi di installazioni e configurazioni. L'utente non paga per il posesso del software, ma per il suo utilizzo. Spesso vengono infatti applicate tariffe in base all'utilizzo del prodotto stesso.

Spesso un'architettura SaaS è multi-tenant, ossia una sola applicazione Server viene utilizzata da più utenti mantenendone, al tempo stesso, separati gli ambienti

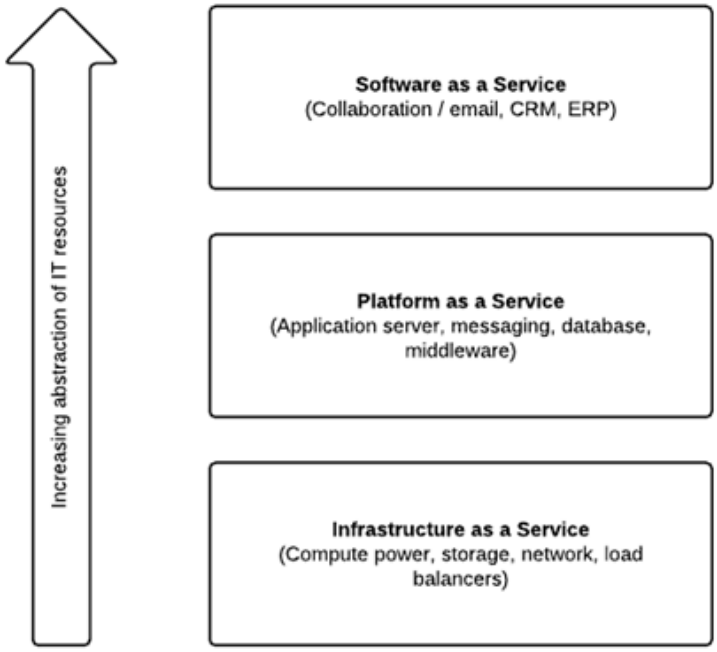

Figura 3.5: Livelli di astrazione nei modelli Cloud Computing

e i dati. Il concetto di multi-tenancy verrà approfondito maggiormente nei prossimi capitoli.

Attualmente le principali software house attive nel mondo aziendale traggono dalle offerte SaaS circa il venti percento del loro fatturato, nel giro di tre-quattro anni la maggior parte di loro prevede che il mercato SaaS diventi sempre più fondamentale.

Vantaggi

Con la metodologia SaaS il provider fornisce il software all'utente già pronto all'uso, l'utente non si deve quindi preoccupare di nient'altro. I clienti, a questo punto, non pagano più per il possesso del software, ma per l'utilizzo del software stesso. In questo modo spesso possono essere notevolmente abbattute le spese iniziali per l'installazione e la configurazione di un prodotto, sostituendole con un costo di fun-

zionamento. Il SaaS comporta quindi una spesa inferiore, ma ricorrente. E' il vendor però a preoccuparsi della manutenzione.

La caratteristica preponderante del cloud SaaS è proprio quella di poter raggiungere i propri dati personali da qualunque luogo e con qualunque dispositivo. E' questa la vera rivoluzione del cloud nel senso più comune del termine. Quando si ha bisogno di lavorare con i propri dati personali, non si ha più la necessità di avere con sè hardware specifico, il proprio laptop o la propria usb key, ma basta avere le proprie credenziali al accedere così allo spazio dedicato.

L'accesso alle tecnologie più sofisticate diviene inoltre alla portata di tutti. Software come gli ERP (Enterprise Resource Planning) e i CRM(Customer Relationship Management), possono essere utilizzati anche da quelle organizzazioni che prima non potevano permettersi un investimento di questa portata.

Un elemento che contraddistingue il SaaS è il notevole grado di apertura verso altre componenti, il che le rende altamente riusabili e flessibili, ciò è ooviamente un grande punto di forza in quanto i requisiti dell'utente spesso cambiano in continuazione.

Gli scenari di aggiornamento cambiano radicalmente. La distribuzione degli aggiornamenti è pressoché immediata e tutti gli utenti hanno la certezza di operare con l'ultima versione del software. Le fasi di upgrading non comportano più l'assenza del servizio come nella versione on premise. L'utente non si accorge neanche del processo di aggiornamento e si ritrova ad utilizzare il software aggiornato.

Svantaggi

Il modello ha ovviamente anche i suoi svantaggi, o perlomeno alcune criticità che vanno tenute presenti. La principale sta nella gestione dei dati aziendali, che sono localizzati nei data center del cloud provider e questo può essere giudicato un rischio per la privacy delle informazioni o addirittura costituire una violazione delle norme che devono osservare le aziende operanti in settori particolari come, ad esempio, la sanità e la difesa.

Tra gli altri aspetti di cui tenere conto ci sono sicuramente le prestazioni e l'affidabilità delle connessioni, che risultano essere i veri bottleneck non solo delle soluzioni SaaS, ma di tutte le soluzioni cloud.

Infine altre criticità riguardano il provider dei servizi. Bisogna tenere conto quanto sia affidabile e se presenta, ad esempio, il rischio che esca dal mercato o ritiri il prodotto software SaaS.

3.3 SOA, Service Oriented Architecture

Il concetto di Service Oriented Architecture è per certi versi affine a quello di SaaS. Un SaaS può rappresentare la percezione da parte dell'utente della modalità di utilizzo di una Architettura Service Oriented. Un servizio ha l'obiettivo di incapsulare una ben precisa funzionalità, per renderla disponibile e accessibile come servizio software sul web. L'Architettura Orientata ai Servizi è quindi uno stile architetturale per la costruzione di una molteplicità di sistemi o applicazioni sulla base della composizione di un insieme di servizi. Spesso, quindi, queste applicazioni non fanno altro che comporre un SaaS, il quale trova in un'architettura di questo tipo una appropriata implementazione.

3.4 Il software On Premise

Abbiamo appena analizzato le peculiarità delle singole tipologie di servizi cloud. Anche il classico modello on premise però ha i suoi punti di forza.

Vantaggi del Software On Premise

- Controllo esclusivo su sistemi e dati
- Alta personalizzazione
- Gestione interna dei dati sensibili
- Alto investimento iniziale ammortizzato nel lungo periodo

Nel caso in cui si abbiano già a disposizione tutte le infrastrutture necessarie, non risulta più vero che la soluzione cloud sia la più economica per far fronte alla necessità di un determinato applicativo. E' anche necessario però che il software sia abbastanza centralizzato per adottare una soluzione on premise.

Occorre quindi fare un'analisi di quanto sia necessario personalizzare il software di cui si ha bisogno e averne il pieno controllo, tenendo conto che un software on premise richiede molta più cura, manutenzione e lavoro di una soluzione basata su cloud. Il paradigma di fornitura on premise risulta ancora essere la soluzione più adatta nel caso in cui la gestione diretta dei dati sia fondamentale per policy aziendali oppure sia necessaria una maggiore flessibilità di configurazione per l'integrazione con altre architetture software. Un'altro requisito che ne può richiedere l'adozione è la necessità che l'architettura fisica del software sia geograficamente centralizzata.

Capitolo 4

SaaS, le tecnologie che ne consentono la realizzazione

4.1 SaaS e i suoi requisiti

4.1.1 Availability

Il concetto di Availability, nel senso generale del termine, è ben definito dallo standard ITU-T E.800: "L'abilità di un sistema di essere in uno stato che soddisfa un determinato requisito, in determinati istanti di tempo, assumendo che le risorse a lui necessarie siano disponibili." Come possiamo vedere, è un concetto ben definito, ad ha quindi le sue metriche ben definite che quantificano l'Availability.

MTTF, Mean Time TO Failure Misura l'intervallo di tempo tra due eventi di "faiulure", in cui il sistema non è riuscito a portare a termine il proprio compito.

Possiamo dire quindi che l'Availability rappresenta la porzione di tempo in cui il sistema si comporta secondo le proprie specifiche. Va tenuto in considerazione anche che, al verificarsi di un fallimento, al tempo di non-Availability si aggiunge il tempo per porre rimedio al fallimento e far ripartire il sistema.

4.1.2 Reliability

La Reliability è definita anch'essa dalla International Telecommunications Union (ITU-T) recommendations E.800, come segue: "L'abilità di un sistema di soddisfare una funzione richiesta, sotto determinate condizioni e per un certo intervallo di tempo

Availability e Reliability nel contesto Cloud Possiamo immaginare, a questo punto, quanto sia fondamentale un'altissima availability per i servizi Cloud. In caso di failure, infatti, possono potenzialmente essere tutti gli utenti serviti dal provider che ha subuto il guasto. I servizi erogati via Cloud dovrebbero essere disponibili da chiunque li richieda e da qualunque parte del mondo ventiquattro ore su ventiquattro. Ovviamente affidabilità massima non è verosimile, ma ci si aspetta una Reliability di molto vicina al 100. Ad esempio, BlueMix dichiara una Reliability del 99,95 percento. Per rendere l'idea, una Reliability del 99,95 percento sta a significare che, sulla base annuale, il servizio non è disponibile per circa 4 ore. La Reliabiity è un concetto affine all'Availability, con la differenza che la Reliability si riferisce all'abilità del sistema di compiere i suoi scopi durante un'intervallo di tempo. Essa infatti si quantifica con una probabilità.

Upgrade Sono gli scenari di Upgrade un'aspetto critico. Come si può immaginare, molti servizi SaaS hanno bisogno di essere continuamente aggiorati e modernizzati. All'uscita di una nuova versione del software occorre che questa venga distribuita a tutti gli utenti del servizio. Distribuire un software su scala Cloud non è semplice come si possa pensare. Non si può infatti interrompere l'erogazione del servizio per far partire il processo di agiornamento del software, che può essere più o meno lungo. Occorrerà quindi adottare delle tecniche che diano l'impressione all'utente di una ontinuità del servizio. Vedremo nei prossimi capitoli quale strategia abbiamo adottato con BigFix SaaS.

Dependability

Capitolo 5

IBM BigFix Detect on SaaS, realizzazione del prototipo

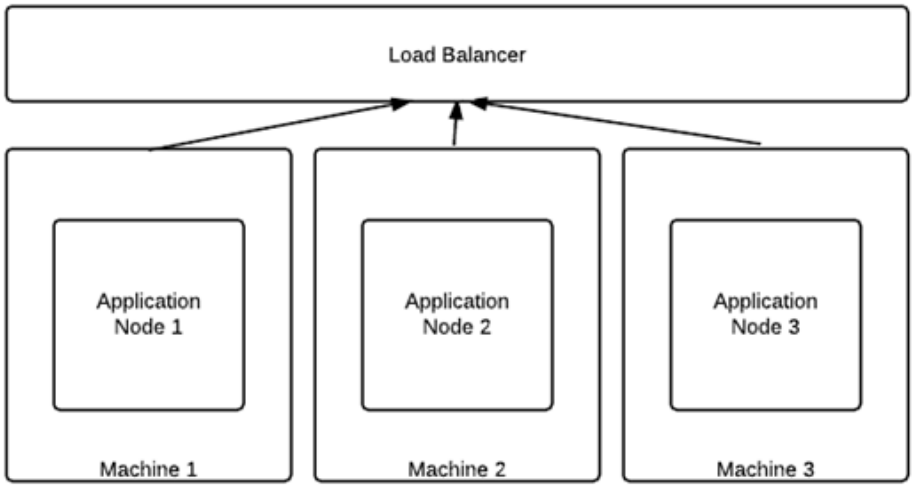

Figura 5.1: Una schematizzazione della distribuzione del carico in una architettura Cloud

Appendice A

Tecnologie Utilizzate(template di prova - ANCORA DA SCRIVERE)

A.1 Linguaggi di programmazione

- PHP 5.4.7
  http://www.php.net/;
- Javascript
  http://www.w3.org/standards/webdesign/script;

A.2 Linguaggi di Markup e Stile

- HTML4/HTML5;
- CSS/CSS3;

A.3 Framework

- Smarty Template Engine
  http://www.smarty.net/;

- JQuery
  http://jquery.com/;
- JQueryUI
  http://jqueryui.com/;
- beContent
  http://www.becontent.org/;

A.4 Ambiente di Sviluppo

A.4.1 Eclipse

Per Eclipse sono state utilizzate due versioni differenti, la 4.2.2 in ambiente Windows e la 3.8.0 in ambiente Ubuntu/Linux
http://www.eclipse.org/
Inoltre è stato utilizzato il pacchetto

- PHP Development Tools 3.1.1
  http://projects.eclipse.org/projects/tools.pdt;

A.4.2 Piattaforma Web

XAMPP

http://www.apachefriends.org

- Apache Web Server ver. 2.4.3
  http://httpd.apache.org/;
- MySql Database Management System ver. 5.5.27
  http://dev.mysql.com/;

A.4.3 Browser Testing

Mozilla Firefox

- Firebug ver 1.11.2
  http://getfirebug.com/
  - Plug-In Validator ver. 0.0.6
    https://addons.mozilla.org/it/firefox/addon/validator/;
  - Plug-In Google Page Speed ver. 2.0.2.3
    https://developers.google.com/speed/pagespeed/?hl=it-IT;

Google Chrome

- Strumenti per gli sviluppatori integrati

Responsive Testing

- Viewport Resizer- Responsive Design Bookmarklet
  http://lab.maltewassermann.com/viewport-resizer/ ;

Dedica a fine pagina