Definizione dei criteri e delle policies per la condivisione e selezione dei dati da condividere all'interno del framework domotico

Vittorio Miori (CNR) – Dario Russo (CNR) – Luca Ferrucci (CNR) – Loredana Pillitteri (CNR)

Breve sommario

Questo documento si concentra sull'analisi delle soluzioni presenti in letteratura per le varie problematiche che insorgono nel momento in cui i dati raccolti dai vari dispositivi del sistema casa devono essere memorizzati ed eventualmente esposti verso infrastrutture esterne.

Parole chiave

Client-Server, P2P, Cloud Computing, data management, data access, data analytics, social IoT, Nuvola It Telecom Italia

Indice

Indice delle figure

Introduzione

Sono state analizzate le architetture Public Cloud, Private Cloud, Hybrid Cloud per poter meglio definire la soluzione adeguata. Telecom Italia ha analizzato alcune architetture di piattaforme (middleware) IoT in Cloud

E' stata svolta anche un'analisi delle possibili architetture Cloud nell'ottica di trovare la soluzione più idonea alla progettazione e successiva implementazione della Piattaforma "Framework Shell". Sono state studiate e analizzate le tipologie di erogazione dei servizi cloud: IaaS, SaaS, PaaS, con lo scopo di identificare e definire i modelli di distribuzione dei servizi dell'infrastruttura da realizzare, e le problematiche a cui si va incontro con l'utilizzo di ciascuno di essi.
Nell'ambito dell'architettura del sistema cooperativo, è stato individuato il Cloud Computing come paradigma computazionale utile per la realizzazione dell'architettura distribuita a supporto del sistema cooperativo.

Partendo da questa analisi, è stata definita la prima Architettura dell'Ambiente Cloud che dovrà ospitare la Piattaforma "Framework SHELL" ed i servizi applicativi che verranno man mano implementati, all'interno della piattaforma Nuvola IT di Telecom Italia. La piattaforma sarà istanziata sull'infrastruttura Cloud di Telecom Italia.

L'architettura ed i servizi offerti da tali piattaforme rispondono molto bene ai requisiti funzionali richiesti dalla Piattaforma oggetto della nostra ricerca. Di seguito si è definita la struttura dati del Power Cloud.
Inoltre è stata avviata la progettazione, la strutturazione, il dimensionamento e la messa in esercizio della piattaforma SaaS sul Cloud di Telecom Italia.

1. Stato dell'arte delle architetture e piattaforme per la condivisione dei dati

1.1 Architetture di condivisione

1.1.1 Architettura Client-Server

Il più diffuso e intrinsecamente distribuito modello di architettura centralizzato è quello tipicamente denominato **client-server.**

Questo complesso logico formato da (uno o più) **server** e (tipicamente molti) **client**, negli anni recenti è stato esteso alla 'interazione tra più macchine o servizi eterogenei, e in tal caso si parla di architettura a 3-tier (o più in generale ad N-tier)

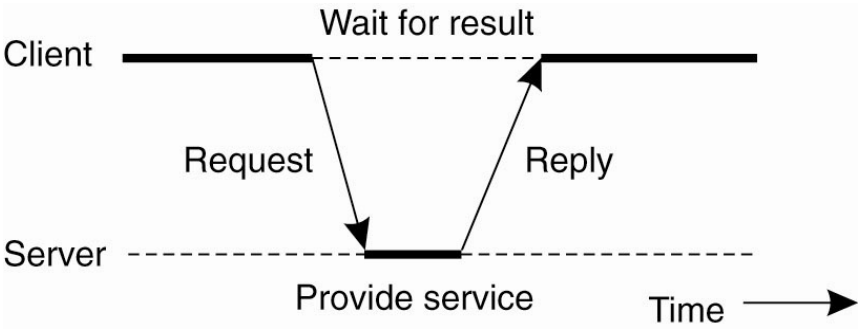

Figura 1: comunicazione client-server

In generale, la comunicazione tra client e server è basata sullo scambio di messaggi e su un comportamento di tipo "request-reply". Vedi *Figura 1.*

Il server riceve dai client delle richieste di servizio e restituisce loro i risultati dopo aver effettuato l'elaborazione relativa. Questo permette di concentrare su una macchina centrale le risorse di elaborazione, il software, la manutenzione, le informazioni critiche o sensibili, gli accorgimenti atti a garantire affidabilità (come i backup). I client, al contrario, possono in generale essere anche macchine con risorse e affidabilità inferiori, che hanno l'unico compito di interagire con l'utente, e che non contengono informazioni critiche o sensibili.

Come precedentemente accennato l'architettura Client-Server (due livelli fisici e perciò definita 2-tier) si è evoluta in un'architettura più complessa, 3-tier, in cui **l'interfaccia utente,** i **processi logico funzionali**, **l'archiviazione informatica dei dati** e l'**accesso ai dati** sono sviluppate e mantenute come moduli indipendenti, la maggior parte delle volte su piattaforme separate.

Oltre ai vantaggi abituali di software modulare con interfacce ben definite, l'architettura 3-tier è destinata a consentire a qualsiasi dei tre livelli di essere aggiornati o sostituiti indipendentemente dal cambiamento di requisiti o tecnologia. Ad esempio, un cambiamento di sistema operativo nel livello di presentazione interesserebbe solo il codice di interfaccia utente.

In genere, l'interfaccia utente viene eseguita su un desktop PC o workstation e utilizza un'interfaccia utente grafica standard. La logica di processo funzionale può essere costituita da uno o più moduli separati in esecuzione su una workstation o da applicazioni server e da un RDBMS residente in un database server o mainframe, contenente i dati di archiviazione logica del computer.
Il livello intermedio può essere anche multi-tier (in questo caso l'architettura complessiva si chiama “n-tier architecture”).

Three-tier architecture ha i seguenti tre livelli:

- ***Livello dell’interfaccia utente (User-interface level)***
  Questo è il livello più alto dell'architettura. Esso mostra le informazioni relative ai servizi utente (come ad esempio merce online, acquisti, contenuti del carrello della spesa).
- ***Livello applicativo (Processing level)***
  Controlla la funzionalità di un'applicazione eseguendo elaborazioni dettagliate.
- ***Livello dei dati (Data level)***
  Questo livello è costituito da server database nel quale le informazioni vengono memorizzate e recuperate. Questo livello mantiene i dati neutrali e indipendenti cioè dalle applicazioni server o dalla logica di business. Fornendo informazioni del proprio livello inoltre migliora la scalabilità e le prestazioni.

Un esempio di quanto precedentemente dichiarato è illustrato nella *Figura 2* e mostra la stratificazione dei livelli in un motore di ricerca web.

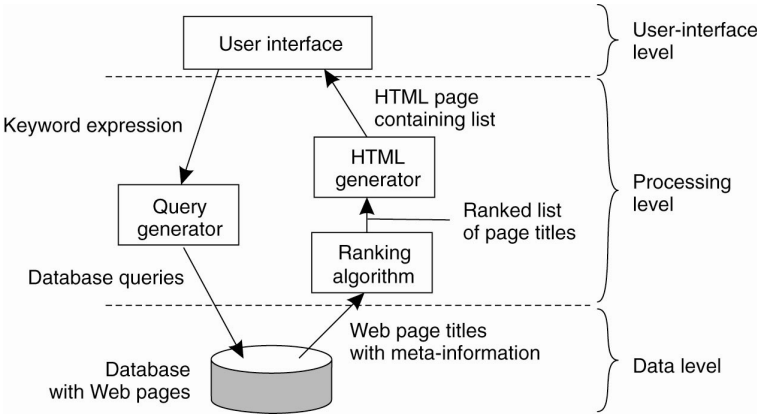

Figura 2: livelli in un motore di ricerca

Nel campo dello sviluppo di applicazioni web, il 3-tier è spesso utilizzato per fare riferimento a website che sono costruiti utilizzando tre livelli:

- **Un front-end web server** che fornisce contenuti statici e, talvolta, alcuni contenuti che risiedono nella *cache* dinamica. In applicazioni web based, il front-end è il contenuto visualizzato dal browser. Il contenuto può essere statico o generato dinamicamente.
- **Un processo di contenuti dinamici** generati a livello di application server, per esempio Java EE, ASP.NET, PHP, Ruby, piattaforma ColdFusion.
- **Un back-end database**, che comprende due serie di dati e il database management system o RDBMS, cioè il software che gestisce e fornisce l'accesso ai dati stessi.

1.1.2 Architettura P2P

Generalmente per architettura **peer-to-peer** (o P2P), si intende una rete di computer o qualsiasi rete informatica che non possiede client o server fissi, ma un numero di nodi equivalenti (peer) che fungono sia da client che da server verso altri nodi della rete, come schematizzato in *Figura 3*. Pertanto è possibile affermare che una rete peer to peer sfrutta la potenza di calcolo dei client connessi ad essa e riduce notevolmente il carico sui server.

Grazie al modello P2P, i computer possono comunicare e condividere i file e altre risorse tra loro senza passare per l'intermediazione di un server centralizzato. Vale a dire che ciascun computer (nodo) è responsabile del passaggio dei dati alle altre macchine e agisce ad un livello gerarchico paritario.

Figura 3: architettura di rete P2P

Nell' architettura P2P, come è ragionevole pensare le connessioni non nascono spontaneamente, ma devono essere richieste da una delle parti in causa. Dopo la connessione, ha inizio la comunicazione tra i nodi che avviene tramite uno scambio di messaggi.

I messaggi servono a:

- segnalare la propria presenza sulla rete
- chiedere una o più risorse
- servire la richiesta di una o più risorse
- trasferire le risorse

La validità dei messaggi è data dal TTL (time to live). Tale valore viene decrementato ad ogni hop, ovvero al passaggio da un nodo all'altro della rete.

Se il TTL > 0 allora il messaggio è valido, altrimenti non viene preso in considerazione.

Riassumendo le caratteristiche dei sistemi P2P sono:

- Tutti i nodi (peer) del sistema hanno identiche capacità e responsabilità (almeno in linea di principio)
    - Nodi indipendenti (autonomi) e localizzati ai bordi (edge) di Internet
- Nessun controllo centralizzato
    - Ogni peer ha funzioni di client e server e condivide risorse e servizi. In questo caso ci si riferisce ad un'entità servent = server + client
    - In alcuni casi, possono essere presenti server centralizzati o nodi con funzionalità diverse rispetto agli altri (supernodi)
- Sistemi altamente distribuiti
    - Il numero di nodi può essere dell'ordine delle centinaia di migliaia
- Nodi altamente dinamici ed autonomi

- Un nodo può entrare o uscire dalla rete P2P in ogni momento

- Operazioni di ingresso/uscita (join/leave) dalla rete
  - Ridondanza delle informazioni

1.1.2.1 Topologie di rete P2P

Nell'architettura P2P sipossono individuare 3 tipologie di rete che andremo ad analizzare di seguito:

- Centralizzata a cluster
- Decentralizzata
- Decentralizzata a cluster

Centralizzata a cluster

Un esempio di rete centralizzata a cluster è Napster. La rete adotta un cluster di server connessi tra di loro (non necessariamente risiedono vicini fisicamente). Ogni client può connettersi, conoscendo l'indirizzo IP, ad uno o a più server del cluster, come schematizzato in *Figura 4*.

Il sistema è efficiente, in quanto i server si preoccupano dell'integrità delle connessioni e mantengono un indice delle risorse condivise dai client. Nonostante ciò, questa topologia, essendo centralizzata, presenta il problema del singolo *punto di fallimento*. La caduta del cluster di server rappresenta la caduta della rete.

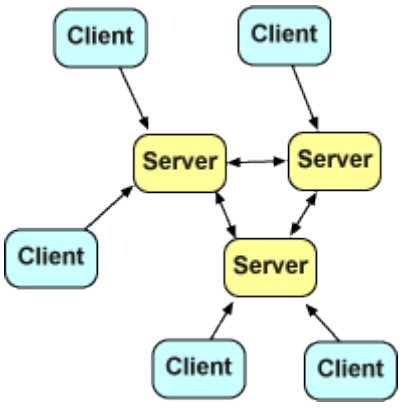

Figura 4: schema di rete Napster

Decentralizzata

Nelle reti decentralizzate non esiste più alcuna gerarchia client-server. Un nodo per connettersi deve conoscere

l'indirizzo IP di un qualsiasi nodo della rete (Figura *5*). Questa topologia elimina il singolo punto di fallimento, ma presenta un problema per la prima connessione: la conoscenza dell'indirizzo IP di un nodo "attivo". Per ovviare a questo problema alcune architetture (Gnutella, Freenet,etc.) adottano una tecnica chiamata ***node caching***, secondo cui viene memorizzata una lista di nodi "attivi", oppure, si avvalgono di veri e propri portali di accesso alla rete.

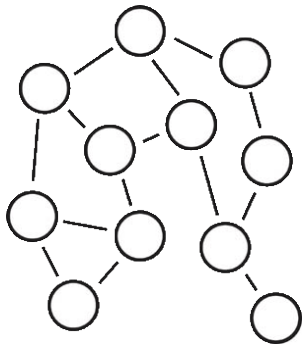

Figura 5: architettura di rete decentralizzata

Decentralizzata a cluster

La novità di questa topologia di rete è la presenza dei "Supernodi", che si comportano da proxy, instradando le varie richieste degli utenti per determinate risorse. Un server centrale, invece, fornisce ai peer gli indirizzi dei Supernodi, necessari per la connessione (*Figura 6*). Un nodo diventa Supernodo nel momento in cui ha una grande quantità di banda e di file condivisi. Le reti decentralizzate a cluster (come Kazaa, Morpheus etc.) sono le più funzionali e hanno avuto maggior successo.

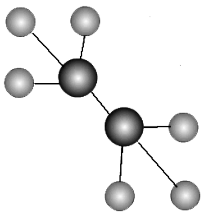

Figura 6: architettura di rete decentralizzata a cluster

1.1.2.2 Ricerca Dati

L'architettura P2P è dotata di un efficiente meccanismo di ricerca che permette agli utenti di localizzare i dati nella rete. Questo meccanismo comprende diversi aspetti:

- **Topologia**: definisce come i nodi sono connessi tra di loro.
  Adottando una rigida topologia (es. centralizzata) si incrementa l'efficienza della rete, ma si restringe l'autonomia dei nodi.
- **Distribuzione risorse**: definisce come i dati sono distribuiti nella rete di utenti. Nelle reti decentralizzate a cluster i dati sono centralizzati in un piccolo numero di nodi, detti supernodi.
- **Routing di messaggi:** definisce come i messaggi sono propagati attraverso la rete. Quando un nodo effettua una richiesta, il messaggio di richiesta viene spedito ad un certo numero di nodi "vicini" (corrispondente al numero di nodi con il quale è connesso). Quando e come i messaggi sono spediti dipende dal protocollo di routing. Spesso il protocollo di routing trae vantaggio dalla particolare topologia di rete e dalla distribuzione delle risorse per ridurre il numero di messaggi spediti.

1.1.2.3 Connessioni

Per comprendere come avviene la ricerca di una risorsa, un aspetto fondamentale in una rete peer to peer è quello di distinguere le diverse connessioni che si stabiliscono tra i nodi della rete. Esistono due diverse connessioni tra i nodi:

- **Connessioni strutturali**
- **Connessioni dirette**

A seconda della connessione che si stabilisce tra i nodi, cambiano le modalità con cui vengono spediti e ricevuti i messaggi.

Le **connessioni strutturali** definiscono gli archi che collegano i nodi di una rete. Queste connessioni servono al routing dei messaggi, cioè a stabilire quale percorso devono seguire i messaggi. Aperta la connessione un nodo spedisce messaggi contenenti la richiesta di una risorsa (file musicali in Napster, di qualsiasi genere negli altri sistemi) ad uno o a più nodi vicini della rete.

Le **connessioni dirette** servono invece a connettere due nodi per il tempo necessario al trasferimento della risorsa. In genere vengono gestite al di fuori della rete, ma alcune architetture (es. Freenet) scelgono di gestirle al suo interno appesantendo inutilmente la rete. I messaggi spediti contengono l'indirizzo IP del nodo mittente e la risorsa richiesta dal nodo destinatario.

1.1.2.4 Sicurezza

A differenza dei sistemi client-sever, dove i server assicurano sicurezza all'intera rete, il sistema P2P non può garantire alti livelli di sicurezza in quanto tale architettura è soggetta ai seguenti problemi:

- **Disponibilità di risorse,**
- **Autenticità ed Integrità dei file**
- **Anonimato**.

Disponibilità di risorse

Ogni nodo in un sistema peer to peer e' in grado di spedire messaggi agli altri nodi e comunicare con loro per contribuire alla disponibilità di risorse della rete. La disponibilità di risorse può portare al verificarsi di possibili attacchi di tipo DOS (Denial of service) i quali hanno lo scopo di occupare la larghezza di banda di un nodo, esaurendo deliberatamente le risorse di un sistema informatico che fornisce un servizio ai client. Ciò può avvenire ad esempio per un sito web su un web server, fino a renderlo non più in grado di erogare il servizio ai client richiedenti.

Autenticità ed integrità dei file

Spesso si pone il problema dell'autenticità e dell'integrità dei file per la comunicazione in rete. L'autenticità è un attributo che ci assicura che un file non sia una copia.

L'integrità è, invece, la caratteristica che ci assicura la non manomissione di un file da parte di qualche nodo. Il problema dell'integrità è risolto con l'utilizzo di funzioni di hash o con la firma digitale. Queste tecniche ci assicurano che un file arrivi al destinatario così come è partito dal mittente.

Sistemi come Freenet utilizzano lo scambio di chiavi per la comunicazione, ottenute con una funzione di hash, SHA-1. Vi sono tre tipi di chiavi:

- KSK: hash con la descrizione del file. Se, però, due file hanno la stessa descrizione avranno anche la stessa chiave, ciò provoca collisioni.
- SSK: chiave che elimina il problema delle collisioni creando un unico **namespace** (coppia utente-file).
- CHK: hash ottenuto con il contenuto del file. Identifica in modo univoco il file.

Anonimato

L'anonimato è un aspetto molto importante per la sicurezza poiché garantisce la privacy. Ogni nodo in questo modo non conosce l'identità degli altri nodi e non conosce i file che possiedono gli altri nodi.

Freenet è il sistema che si sofferma maggiormente su questo aspetto in quanto, a differenza degli altri sistemi, ogni nodo possiede i propri file e una copia dei file più richiesti. Con questa distribuzione delle risorse è difficile risalire a chi sia il proprietario effettivo del file.

1.1.3 Architettura Cloud-Computing

Il Gruppo di TIM, in questo semestre ha avviato lo studio e dell'analisi dello stato dell'arte del Cloud Computing, quale paradigma utile alla progettazione e implementazione, da parte degli altri partner del progetto, della Piattaforma "Framework SHELL".

Il Cloud Computing è un paradigma di calcolo distribuito che fornisce risorse ridondanti, economiche, scalabili in modo dinamico, virtualizzate e accessibili tramite un'interfaccia a servizi (as a service) su Internet.

Fin dal 2006, quando Amazon propose i suoi web services (Amazon Web Services), questa tecnologia è stata utilizzata in diversi ambiti con notevoli vantaggi per utenti pubblici e privati. Le risorse che un sistema Cloud fornisce agli users sono CPU, memorie, reti, sistemi operativi, middleware, applicazioni di alto livello.

Le funzionalità e i servizi offerti per un generico utente da un sistema Cloud sono molteplici. Ad esempio, uno user ha la possibilità di affittare dei servizi di elaborazione, aumentando e diminuendo dinamicamente la capacità di calcolo utilizzata, così da ottenere una determinata qualità del servizio. Inoltre, un utente può sfruttare il Cloud come unità di memorizzazione per conservare grandi quantità di dati, il cui accesso può essere autorizzato solo a specifici utenti o applicazioni.

Un altro aspetto interessante e innovativo dei sistemi Cloud è l'adozione del sistema pay per use, ovvero si pagano solo le risorse effettivamente utilizzate (per esempio ore CPU, banda di rete, quantità di storage, uso del software, applicazioni). Così i costi del servizio sono proporzionali all'utilizzo reale. Come detto in precedenza, gli utenti non hanno conoscenza o controllo sull'infrastruttura tecnologica, hardware e software, che supporta i loro usi e, quindi, non devono preoccuparsi di eventuali problemi tecnici di basso livello e dei dettagli della loro gestione.

Un meccanismo software fondamentale nella realizzazione di un sistema Cloud è la virtualizzazione. Una macchina virtuale è un contenitore software, totalmente isolato da altre macchine virtuali, che può eseguire il proprio sistema operativo e le proprie applicazioni come se fosse un singolo computer fisico. La virtualizzazione consente di eseguire più macchine virtuali su una singola macchina fisica oppure una macchina virtuale su più macchine fisiche, laddove ogni macchina virtuale condivide (con le altre) le risorse del computer fisico su cui è allocata. In questo modo sullo stesso host ogni macchina virtuale esegue il proprio sistema operativo e le proprie applicazioni, sfruttando le risorse hardware che le sono state assegnate.

Un ulteriore vantaggio della virtualizzazione consiste nella possibilità di poter migrare una macchina virtuale da un server ad un altro, in modo trasparente per l'utente.

Nella letteratura scientifica ed industriale esistono varie definizioni [14, 13, 8, 17, 6, 16, 11] di Cloud Computing,

ma dal punto di vista formale non esiste una definizione universale e univoca. Di seguito sono riportate le principali definizioni:

1. Il Cloud Computing è un modello per abilitare l'accesso via rete e su richiesta a un pool condiviso di risorse di calcolo configurabili (per esempio reti, servers, dispositivi di memorizzazioni, applicazioni e servizi) che possono essere rapidamente fornite e rilasciate con minimo sforzo gestionale o interazione con il fornitore di servizi. Questo modello Cloud è composto da cinque caratteristiche essenziali, tre modelli di servizio e quattro modelli di deployment. (National Institute of Standards and Technology (NIST)) [14];
2. Il Cloud è un ambiente di esecuzione elastico delle risorse che coinvolge più stakeholders e che fornisce un servizio commisurato a granularità multipla per il livello di qualità specificata (del servizio). (dal report "The Future of Cloud Computing" della EU) [13]
3. Gartner definisce il Cloud Computing come uno stile di calcolo in cui vengono fornite capacità IT scalabili ed elastiche sotto forma di servizi utilizzando la tecnologia Internet [8].

È possibile fare una sintesi delle varie definizioni e definire il paradigma Cloud Computing come un sistema di elaborazione che fornisce risorse hardware (capacità di calcolo, di comunicazione e di memorizzazione) e software (sistemi operativi, ambienti di sviluppo, programmi applicativi, ecc.) agli utenti che ne fanno richiesta. Gli utenti richiedono i servizi di un sistema Cloud tramite un'interfaccia Web, senza avere conoscenza o controllo sull'infrastruttura tecnologica che fornisce i servizi richiesti. La fornitura di tali servizi avviene in modo organizzato, scalabile ed elastico. Dal punto di vista fisico/infrastrutturale un sistema Cloud viene realizzato su uno o più data center, ovvero un insieme di server su cui vengono eseguite le applicazioni e memorizzati i dati.
In *Figura 7* si riporta uno schema della definizione di Cloud Computing, in cui si evidenziano le cinque caratteristiche essenziali, i tre modelli di servizi e i quattro modelli di deployment.

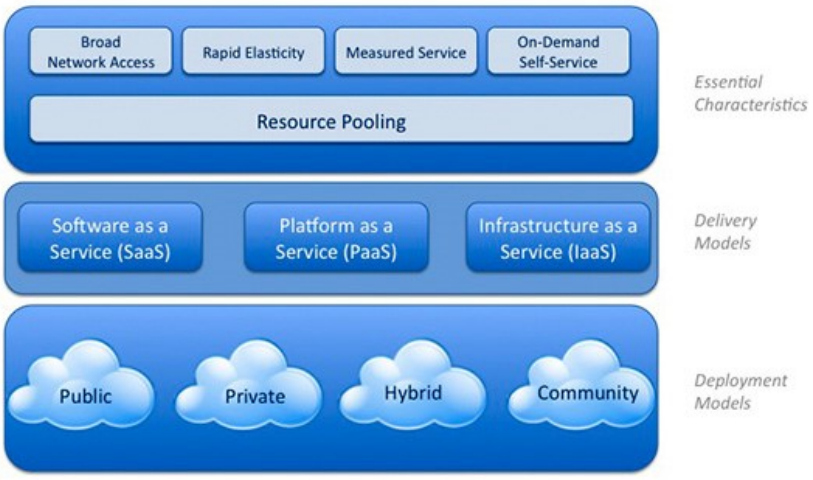

Figura 7: Definizione di Cloud Computing

Il modello Cloud si contraddistingue per cinque caratteristiche essenziali [14]:

- **On–demand** self–service. Un cliente può unilateralmente fare richiesta e ottenere risorse (computazionali, di memorizzazione o altro) per svolgere i suoi task, senza richiedere l'intervento umano dei fornitori dei servizi stessi;
- **Broad network access**. Le risorse sono disponibili attraverso la rete mediante meccanismi standard che rendono flessibile l'utilizzo di piattaforme client eterogenee sia leggere che complesse (per esempio dispositivi mobili, tablets, notebook e workstations) e vengono accedute tramite un'Interfaccia a Servizi;
- **Resource pooling.** Le risorse di calcolo del fornitore vengono organizzate per servire più clienti, utilizzando il modello multi-tenant, in cui le risorse fisiche e virtuali sono assegnate dinamicamente e riassegnate a seconda della richiesta dei clienti. Le risorse offerte sono indipendenti dalla loro locazione fisica, ovvero il cliente generalmente non ha conoscenza dell'esatta locazione fisica delle risorse a lui fornite. Tuttavia, il fornitore potrebbe permettere all'utente di specificare dei vincoli sulla locazione fisica delle risorse a lui assegnate a un livello di astrazione alto (ad esempio continente, nazione, oppure data center);
- **Rapid elasticity.** Le risorse possono essere fornite rapidamente ed elasticamente, permettendo un incremento e un decremento dinamico della capacità computazionale a disposizione dell'utente, in base alle necessità dei task richiesti. Dal punto di vista dell'utente, le risorse disponibili appaiono illimitate, così da poter essere richieste in qualsiasi quantità e in qualsiasi momento;
- **Measured service.** I sistemi Cloud controllano e ottimizzano automaticamente l'utilizzo delle risorse, sfruttando la capacità di misurare l'utilizzo delle risorse al livello necessario per il tipo di servizio (per esempio servizi di memorizzazione, di calcolo, banda di comunicazione e account utente attivi).
  Il monitoraggio dell'utilizzo dei servizi è molto importante, perché permette al fornitore di reagire a eventuali picchi di richiesta allo scopo di garantire al cliente la Qualità del Servizio (QoS) promessa.
  L'utilizzo delle risorse può essere monitorato e riportato in modo trasparente sia per il fornitore sia per il cliente.

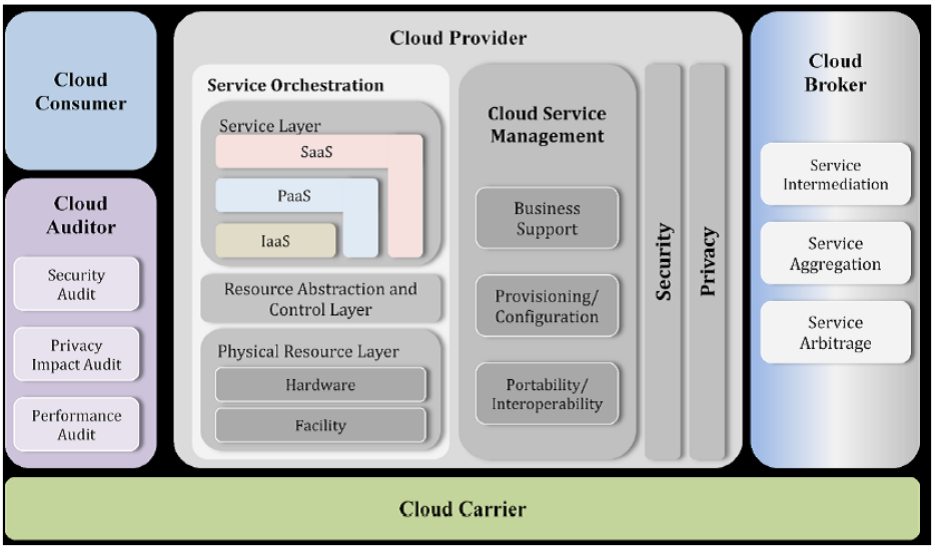

Figura 8: Modello concettuale per il Cloud Computing secondo il NIST

In base a questo modello concettuale (vedi *Figura 8*), è possibile individuare cinque ruoli principali all'interno dell'architettura:

- **Cloud Consumer (o Cloud Service Consumer)**, ovvero il soggetto che utilizza i servizi cloud. Nello specifico, si tratta di un utente o di un'organizzazione che esamina il catalogo dei servizi di un Cloud Provider, richiede servizi specifici e li utilizza. Inoltre, utilizza degli accordi sui livelli di servizio, Service Level Agreements (SLAs), per specificare i requisiti sulle prestazioni tecniche che devono essere soddisfatti da un Cloud Provider;
- **Cloud Auditor,** ovvero il soggetto che può valutare i servizi da un Cloud Provider dopo aver eseguito un esame indipendente sui servizi stessi, come i controlli per la sicurezza, l'impatto sulla privacy e sulle prestazioni;
- **Cloud Provider (o Cloud Service Provider),** ovvero il soggetto che permette l'utilizzazione del servizio alle parti terze interessate. In pratica, acquisisce e gestisce le infrastrutture elaborative necessarie per la fornitura dei servizi, assicura l'esecuzione dei programmi che consentono i servizi stessi e le infrastrutture per l'erogazione dei servizi tramite rete. Senza dimenticare le attività riguardanti la sicurezza e la privacy;
- **Cloud Broker,** ovvero il soggetto che gestisce l'impiego, le prestazioni e l'erogazione dei servizi cloud e cura le relazioni tra un Cloud Consumer e un Cloud Provider. In particolare, un Cloud Broker opera in tre aree:
  - intermediazione, ovvero estende un servizio Cloud fornendo servizi a valore aggiunto agli utenti, come la gestione dell'accesso, della sicurezza e dell'identità;

- aggregazione, ovvero combina e integra servizi diversi in un servizio nuovo, assicurando l'integrazione e la sicurezza nei trasferimenti di dati tra il Cloud Consumer e i differenti Cloud Provider;
- arbitraggio, ovvero sceglie i servizi cloud da fornitori diversi in modo flessibile e dinamico, tenendo conto sia di criteri economici sia di disponibilità;

- **Cloud Carrier**, ovvero il soggetto che agisce da intermediario per fornire la connettività e il trasporto di servizi Cloud tra il Cloud Consumer e il Cloud Provider. Il Cloud Carrier fornisce l'accesso al Cloud Consumer tramite le reti e i dispositivi di accesso.

1.1.3.1 Modelli a Servizi

I servizi Cloud possono essere classificati in tre modelli principali, quali *Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS)*, ovvero in base al tipo di risorse fornite dagli stessi. I servizi sono gestiti da un livello di controllo che, a sua volta, si appoggia sulle risorse fisiche messe a disposizione dal fornitore, a prescindere dal tipo di modello adottato.

Nelle tre sottosezioni seguenti saranno descritte le peculiarità dei tre modelli.

Infrastructure as a Service

Nel modello *Infrastructure as a Service (IaaS)*, il servizio offerto consiste in una infrastruttura di elaborazione che comprende server, tipicamente virtualizzati, con specifiche capacità computazionali e/o di memorizzazione. L'utente può controllare tutte le risorse di storage, i sistemi operativi e le applicazioni deployate.
Mentre, l'utente stesso ha generalmente un controllo limitato sulle impostazioni di rete, come la configurazione dei firewall.
Uno user può richiedere al provider un insieme di macchine virtuali, sfruttando le immagini standard fornite dal servizio oppure richiedendo la creazione di immagini personalizzate.
Inoltre, l'utente può richiedere che le macchine virtuali assegnate siano connesse tra di loro attraverso una rete virtuale, sia con indirizzi pubblici sia con indirizzi privati.
La gestione delle macchine virtuali avviene tramite l'interfaccia offerta dal fornitore del servizio, che tipicamente permette di incrementare o diminuire la capacità delle macchine esistenti, di rilasciarne alcune, oppure di crearne delle nuove.
L'utente deve inoltre gestire il sistema operativo delle macchine a lui assegnate e delle applicazioni installate.
Nel caso di servizio di memorizzazione, l'utente può richiedere uno spazio di memorizzazione per caricarvi i suoi dati e, successivamente, può aumentarlo o ridurlo a seconda delle sue esigenze.
Un esempio di fornitore di servizi IaaS di tipo computazionale è Amazon Elastic Compute Cloud (EC2), che permette di creare macchine virtuali, standard o personalizzate, e di gestire le stesse durante la loro esecuzione.

Un esempio di servizi IaaS di memorizzazione è Amazon Simple Storage Service (S3), che permette di memorizzare i dati e di accedervi da qualsiasi locazione e in qualsiasi momento tramite una semplice interfaccia web.

Platform as a Service

Nel modello di servizio *Platform as a Service (PaaS)* il fornitore dello stesso permette lo sviluppo, l'installazione e l'esecuzione di applicazioni sviluppate dall'utente sulla sua infrastruttura.
Naturalmente, le applicazioni devono essere create usando linguaggi di programmazione, librerie, servizi e strumenti supportati dal fornitore, che costituiscono la piattaforma di sviluppo fornita come servizio.
Può essere fornita all'utente un'apposita interfaccia di programmazione (API) per lo sviluppo di queste applicazioni. Altresì, può essere prevista la compatibilità con un insieme di linguaggi, librerie e strumenti con cui l'utente può scrivere le applicazioni, interagenti con il servizio stesso.
Le specifiche funzionalità dell'API dipendono dal servizio offerto. Così come la loro esecuzione è assicurata dal fornitore del servizio stesso.
Il client non ha alcun controllo sull'infrastruttura, ma gestisce completamente le applicazioni implementate e ha la possibilità di configurare l'ambiente che ospita l'applicazione stessa.
Un esempio di servizi Cloud di tipo PaaS è Google Apps Engine, che permette di sviluppare applicazioni sia in Java sia in Phyton, fornendo per entrambi i linguaggi di programmazione il Software Development Kit e un plugin per utilizzare l'ambiente di sviluppo Eclipse.
Un altro esempio di servizio Cloud di tipo PaaS è dato da Microsoft Azure, che fornisce le piattaforme e gli ambienti di sviluppo di Microsoft, come .NET, C#.

Software as a Service

Nel modello di servizio *Software as a Service (SaaS)* sono fornite agli utenti applicazioni software, che possono essere utilizzate su richiesta.
Le applicazioni sono accessibili dai dispositivi client attraverso un'interfaccia, come un browser web. In questo caso, l'utente non ha nessun tipo di controllo o di accesso all'infrastruttura Cloud, alla rete, al sistema operativo e ai server.
È possibile, però, fare delle modifiche limitate alle impostazioni delle applicazioni.
In genere, questi servizi Cloud di tipo SaaS sono implementati sui servizi Cloud di livello inferiore (PaaS e IaaS).
Esempi di applicazioni di servizi Cloud di tipo SaaS sono le Google Apps, che offrono tramite il web, collegandosi con un account Google, di accedere a un insieme completo di applicazioni.
Esistono applicazioni per la gestione della posta elettronica, Gmail, per l'archiviazione dei dati, Google Drive,

per l'editing di testi, di fogli di calcolo e di presentazioni, Google Docs, per la gestione degli appuntamenti e degli eventi, Google Calendar, per la navigazione, Google Maps.
Un altro esempio è fornito da Microsoft con la suite Office sul proprio Cloud secondo il modello SaaS.

1.1.3.2 Esempi di Sistemi Cloud

In ambito accademico e industriale esistono diversi sistemi e servizi Cloud. In questa sezione, verranno descritti brevemente i principali sistemi e servizi Cloud disponibili.

Sistemi Cloud Pubblici

Amazon, Google e Microsoft sono i principali providers di sistemi di Public Cloud e con i loro prodotti rappresentano un punto di riferimento globale.
Amazon è uno dei primi fornitori di servizi Cloud ed è uno dei più conosciuti e utilizzati. L'offerta *Cloud di Amazon*, *Amazon Web Services (AWS)* [1], si compone di vari servizi, i più rilevanti sono *Amazon EC2* e *Amazon S3*.
Amazon EC2, *Amazon Elastic Compute Cloud* [2], è un servizio Web che fornisce capacità di elaborazione informatica nel Cloud dimensionabili secondo necessità. È un servizio Cloud di tipo IaaS, in cui i servizi possono essere richiesti, configurati e controllati durante la loro esecuzione tramite una semplice interfaccia di tipo Web service. Inoltre, Amazon EC2 fornisce un ambiente di calcolo virtuale, che permette di lanciare il numero desiderato di istanze di macchine virtuali con diversi sistemi operativi, di caricare su queste macchine un ambiente personalizzato, di gestire i permessi di accesso della rete e di eseguire l'immagine utilizzando il numero di macchine richiesto.
Il tempo necessario per ottenere e avviare nuove istanze di macchine virtuali è ridotto ed è possibile variare rapidamente la capacità computazionale richiesta, aumentandola o diminuendola a seconda delle esigenze di calcolo. Amazon EC2 offre agli sviluppatori strumenti per creare applicazioni resistenti agli errori.
Per quanto concerne la sicurezza e l'affidabilità, Amazon EC2 funziona in combinazione con Amazon VPC (Virtual Private Cloud), dove le istanze di calcolo sono ubicate con un range di IP specificati per l'utente.
In più, è possibile effettuare il provisioning di risorse EC2 su host dedicati o come istanze dedicate.
Esistono vari tipi di istanze, dimensionate per varie tipologie di esigenze computazionali, ovvero diverse combinazioni di capacità di CPU, di memoria, di storage e di rete.
Ad esempio, la "Small Instance" è una piattaforma a 32 bit con 1.7 GB di memoria, 1 EC2 Compute Unit (ovvero 1 virtual core con 1 EC2 Compute Unit) e 160 GB di spazio disco locale. Mentre, la "Extra Large Instance" è una piattaforma a 64 bit con 15 GB di memoria, 8 EC2 Compute Units (ovvero 4 virtual cores con 2 EC2 Compute Units ciascuno) e 1690 GB di spazio disco locale. Amazon EC2 si integra perfettamente con gli altri servizi forniti da Amazon, quali Amazon Simple Storage Service (Amazon S3), Amazon Relational

Database Service (Amazon RDS), Amazon SimpleDB e Amazon Simple Queue Service (Amazon SQS), per fornire una soluzione completa per il calcolo, la gestione di query e la memorizzazione dati.
Amazon S3, *Amazon Simple Storage Service* [3], è un servizio che offre capacità di memorizzazione per oggetti sicura, durevole e altamente scalabile su Cloud.
Attualmente, tramite un'interfaccia di tipo Web service, è possibile memorizzare dati su Amazon S3 e successivamente fare interrogazioni o semplici consultazioni. P
er garantire la sicurezza dei dati, sono previsti meccanismi di autenticazione così da proteggere da accessi non autorizzati e vengono applicate tecniche crittografiche per proteggere i dati a riposo e durante i trasferimenti tramite rete. Amazon S3 offre diverse classi di storage progettate per differenti casi d'uso, tra cui: Amazon S3 Standard per lo storage generico di dati con accessi frequenti, Amazon S3 Standard–Infrequent Access (Standard–IA) per lo storage di dati con minore frequenza di accesso e Amazon Glacier per l'archiviazione a lungo termine. Come per Amazon EC2, Amazon S3 può essere utilizzato come servizio indipendente o insieme ad altri servizi Amazon Web Services, come lo stesso Amazon Elastic Compute Cloud, AWS Identity and Access Management, nonché gateway e servizi di trasferimento.
Google Apps è un servizio su abbonamento di Google, che fornisce servizi di Cloud Computing a organizzazioni e aziende.
I servizi offerti sono gli stessi offerti da Google per gli utenti finali, come Gmail, Drive, Hangouts, Calendar, Google Docs e altro ancora.
Attualmente, più di cinque milioni di utenze business si sono spostate su Google Apps, così da permettere ai propri dipendenti di lavorare con una maggiore flessibilità ed essere più produttivi, in maniera indipendente dal luogo in cui si trovano [10].
I servizi di Google Apps sono completamente Web-based, non ci sono, quindi, server da acquistare e da mantenere, in modo da ridurre i costi IT e la complessità della loro gestione. Google Apps include strumenti sicuri ed efficaci per le e-mail, per le chat, per la gestione dei calendari, per la creazione di documenti, per le condivisioni di video e per la progettazione di siti web. Inoltre, passare a Google Apps è reso semplice dagli strumenti di migrazione messi a punto per tutti i sistemi legacy esistenti. In particolare, le app più importanti offerte sono:

- *Gmail.* Fornisce 30 GB di memorizzazione per utente, indirizzi mail personalizzati, una piattaforma avanzata per lo spam filtering e un accesso personalizzato e sicuro da diversi dispositivi mobili (come Android, iOS, BlackBerry, ecc.);
- *Google Calendar.* Permette agli utenti di organizzare i loro impegni e di creare e gestire calendari condivisi, che si integrano facilmente con Gmail, Drive, Contatti, Sites e Hangouts;

- *Google Docs.* Abilita la collaborazione real-time su documenti, fogli di calcolo, presentazioni e disegni. In questo modo, i gruppi di lavoro riescono a collaborare in modo più efficiente ed evitano inconvenienti dovuti al controllo del versioning e all'uso di attachment molto pesanti;
- *Google Sites*. Consente di creare e condividere i propri progetti di siti web, senza avere particolari competenze tecniche.

Le aziende possono beneficiare delle Google Apps per migliorare la propria produttività, così da evitare costi e complessità delle soluzioni legacy, collaborare real-time su documenti o ricercare velocemente messaggi in Gmail. Ma possono anche avere una maggiore garanzia su un'adeguata gestione della sicurezza e dell'affidabilità.

Le reti di data center Google, infatti, sono progettate per essere sicure, affidabili e ridondanti. Per esempio, Google Apps prevede la cosiddetta "verifica a 2 passi", un sistema di autenticazione che aumenta sensibilmente la protezione rispetto ad accessi indesiderati.

Anche la protezione dei dati è sicura e compliant rispetto agli standard SSAE 16/ISAE 3402 Type II audit.

Microsoft fornisce diversi servizi Cloud, tra i quali i più importanti sono *Windows Azure Compute, SQL Azure Database e Windows Azure Storage*.

Windows Azure Compute [18] è un insieme di servizi Cloud integrati di analisi, database, elaborazione, archiviazione, applicazioni per dispositivi mobili e web in continua evoluzione.

Gli strumenti integrati, i modelli predefiniti e i servizi gestiti consentono di creare e gestire app aziendali, per dispositivi mobili, Web e Internet of Things in modo più semplice e più rapido, usando le proprie competenze e le tecnologie note. Windows Azure Compute fornisce una piattaforma Cloud di tipo IaaS e PaaS, tramite la quale è possibile eseguire e gestire applicazioni nei centri di elaborazioni dati di Microsoft. Un'applicazione di Windows Azure è basata su uno o più componenti denominati "ruoli".

Esistono tre diversi tipi di ruoli: Web, Worker e Virtual Machine (VM). Il ruolo Web viene utilizzato per ospitare le applicazioni Web front-end in Internet Information Services (IIS).

Il ruolo Worker viene utilizzato principalmente per ospitare l'elaborazione eseguita in background dietro un ruolo Web. In genere un'applicazione interagisce con gli utenti tramite un ruolo Web e affida l'elaborazione delle attività a un ruolo Worker.

Infine, il ruolo Virtual Machine consente di eseguire un'immagine personalizzata di Windows Server 2008 R2 (Enterprise o Standard) su Windows Azure.

L'immagine consiste in un file che rappresenta un hard disk virtuale (VHD) su cui è installato Windows Server 2008 R2.

Microsoft SQL Azure Database [18] fornisce un database scalabile e a disponibilità piuttosto elevata, basato sulle tecnologie SQL Server.

SQL Azure utilizza il modello relazionale basato su T-SQL e gli stessi strumenti di sviluppo e gestione già usati per i database locali.

È possibile eseguire operazioni di query centralizzate, come la creazione di report e l'estrazione di dati da più database, ottenendo un unico set di risultati. Il database SQL offre un portfolio di funzionalità di sicurezza che aiutano a soddisfare al meglio i criteri di conformità dell'organizzazione o richiesti dal settore.
Sono anche implementati criteri a livello di database per limitare l'accesso ai dati sensibili con sicurezza a livello di riga, maschera dati dinamica e Transparent Data Encryption per il database SQL di Azure.
Il database SQL è inoltre verificato anche da revisori primari del cloud come parte delle certificazioni di conformità e delle approvazioni chiave di Azure, come il contratto di società in affari HIPAA, ISO/IEC 27001:2005, FedRAMP e le clausole modello UE.
Windows Azure Storage [18] offre servizi di archiviazione di base protetti, scalabili e facilmente accessibili, che assicurano un'archiviazione permanente e duratura nel Cloud.
Sono disponibili come parte dell'account di archiviazione di Windows Azure: i Binary Large Objects (BLOB), Azure Blob Storage, generalmente usati per dati non strutturati, come documenti e media files; le tabelle, Azure Table Storage, usate per dati strutturati no–SQL; le code, Azure Queue Storage, per archiviare i messaggi in modo affidabile.
Diversamente dall'archiviazione locale, i blob, le tabelle e le code sono accessibili da più applicazioni o istanze di applicazioni contemporaneamente e rappresentano un'archiviazione dedicata anziché temporanea.
Inoltre, Windows Azure Storage prevede un’archiviazione con ridondanza geografica a centinaia di chilometri, prestazioni rapide, condivisione di file SMB standard di settore in macchine virtuali, supporto per REST, .NET, Java, C++, node.js, PowerShell.
Nuvola di Telecom Italia e tutte le atre piattaforme cloud sono rese disponibili con una serie di servizi accessori.
Ad esempio, il servizio Nuvola IT Self Data Center nasce per fornire un accesso rapido, sicuro, con accesso a larga banda, a risorse computazionali con la massima flessibilità, scalabilità e modularità di configurazione delle stesse.
Il Servizio Self Data Center mette a disposizione una porzione di risorse della Nuvola Italiana (Data Center di TIM), che può essere gestito, configurato in autonomia.
Il servizio viene erogato su una piattaforma multitenant basata su suite di prodotti di virtualizzazione VMware.
L’utilizzatore può in qualsiasi momento usufruire dell’accesso alla server farm, utilizzare i servizi di supporto e gli strumenti di self management. Il servizio viene erogato in modalità Allocation Pool che prevede la disponibilità iniziale di un determinato asset di risorse elaborative riservate allo sviluppo ed all’esercizio della piattaforma Domus Energia (stabilite in fase progettuale), con la possibilità di superare la soglia prestabilita di GHz e RAM (vCore e vRam) per un ulteriore 33% in full sharing (risorse non riservate) per garantire la gestione di picchi di elaborazione del proprio Data Center Virtuale.

Sistemi Cloud Privati

Un utente, che deve installare un sistema Cloud nel proprio data center, può scegliere tra diversi strumenti

software disponibili.

Alcuni di questi software sono gratuiti e open-source, altri sono prodotti commerciali e soggetti al pagamento di una licenza. Mentre, altri hanno sia una versione gratuita, che l'utente può utilizzare per testare il sistema, sia una versione commerciale.

I più diffusi strumenti software per progettare Private Cloud sono *OpenNebula* ed *Eucalyptus*.

OpenNebula [15] è un progetto nato con l'obiettivo di definire uno strumento open source di riferimento per la gestione di piattaforme per il Cloud Computing.

Tale progetto è in grado di gestire centri di elaborazione dati, che possono essere eterogenei e distribuiti. OpenNebula è uno standard industriale per il Cloud Computing di tipo IaaS, che offre una soluzione completa per la gestione di centri elaborazione dati virtualizzati per la creazione di Cloud privati, pubblici e ibridi.

OpenNebula supporta i più comuni strumenti di virtualizzazione per la gestione delle macchine virtuali, quali Xen, KVM e VMware.

Per semplificare la creazione di macchine virtuali, OpenNebula fornisce un repository per le immagini, che permette di creare e condividere immagini di macchine virtuali e un repository per i template, che permette di registrare definizioni di macchine virtuali, così da essere utilizzate successivamente per creare nuove macchine virtuali.

Una volta che un template è stato istanziato in una macchina virtuale, un certo numero di operazioni possono essere fatte per controllarne il ciclo di vita, come arrestarla, come sospenderla, come riattivarla, oppure come migrare la stessa da un centro di elaborazione dati a un altro.

Dal punto di vista della sicurezza, OpenNebula supporta account di singoli utenti o di gruppi, così come vari meccanismi di autenticazione e autorizzazione.

Queste caratteristiche possono essere utilizzate per creare compartimenti isolati all'interno della stessa Cloud. In più, OpenNebula fornisce un potente meccanismo di sicurezza, la lista di controllo degli accessi, che supporta la gestione dei ruoli e la gestione dei permessi a grana fine.

Eucalyptus [7] è un software open source per installare sistemi Cloud di tipo IaaS, che sfrutta l'infrastruttura IT già esistente in un centro di calcolo. Eucalyptus, acronimo di Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems, è il risultato di un progetto del dipartimento di informatica della University of California.

È basato sul sistema operativo Linux e sulla tecnologia web service e supporta diverse tecnologie di virtualizzazione, permettendo di creare sistemi Cloud senza bisogno di hardware particolare o di configurazioni specifiche.

L'interfaccia utente per creare e gestire le macchine virtuali è la stessa di Amazon EC2. Quindi, l'utente di Eucalyptus può usare gli stessi tools utilizzati per interagire con i servizi Cloud forniti da Amazon.

Queste caratteristiche di Eucalyptus lo rendono particolarmente idoneo all'istallazione di Cloud Ibridi, per combinare l'utilizzo di risorse private con risorse ottenute da fornitori Cloud Pubblici.

1.2 Piattaforme di condivisione

In questa sezione verrà presentata un'analisi di diverse le piattaforme IoT disponibili attualmente sul mercato, sia proprietarie che open-source, che permettono di collegare oggetti intelligenti a "Internet".

L'elenco delle 39 piattaforme in esame non è affatto esaustivo ma identificano un campione rappresentativo delle piattaforme disponibili.

La *Tabella 1* elenca le piattaforme analizzate e riassume alcune caratteristiche identificate come fondamentali per soddisfare le esigenze e le specifiche di progetto al fine di poter scegliere la piattaforma IoT più appropriata da implementare nell'ambiente SHELL.
In particolare, il colore verde indica che una caratteristica di una particolare piattaforma si adatta alle specifiche di progetto, mentre il colore rosso indica una mancata corrispondenza tra la caratteristica delle piattaforme e le specifiche di progetto. È stato aggiunto un colore arancione intermedio per indicare una corrispondenza parziale.

Per una migliore lettura della tabella 1 si riporta quanto segue:

- la *colonna (a)* elenca i tipi di dispositivi supportati dalla piattaforma. Le piattaforme che richiedono un gateway specifico per la connessione di dispositivi IoT dipendono dai fornitori di quella specifica piattaforma limitando così la reattività della stessa ad adottare nuovi protocolli e il numero crescente di dispositivi IoT eterogenei.
- La *colonna (b)* descrive il tipo di piattaforma IoT. Nella maggior parte dei casi, le piattaforme vengono fornite in cloud in una forma PaaS o in un servizio SaaS. In particolare il PaaS si riferisce alle piattaforme che forniscono servizi di cloud computing per dispositivi e dati IoT, i quali comprendono servizi di storage, gestione dei dispositivi, connettività dei dispositivi, meccanismi di backup e supporto online. Al contrario, SaaS si focalizza sul mashup dei dati che utilizzano le capacità di cloud computing. E' stato inoltre aggiunto un tag aggiuntivo, Machine-to-Machine (M2M), se la piattaforma mira principalmente a questa tipologia di IoT.
- Il tipo di architettura è indicato nella *colonna (c)*.
- La tabella, alla *colonna (e),* include anche informazioni sulla disponibilità di un' API REST (Transparency State Transposing), nonché i meccanismi di controllo di accesso ai dati, *colonna (f)* e di rilevazione dei servizi, *colonna (g)*. Nello specifico si è verificato come solo alcune piattaforme non dispongono di un'API REST e questo dimostra che i servizi attuali di IoT tenderanno a diventare sempre più come i servizi web tradizionali (cioè il Web of Thinghs). In particolare, i mashup dei servizi di IoT e

l'analisi dei dati saranno integratori chiave per il futuro delle tecnologie IoT.

- Da sottolineare inoltre che alcune piattaforme open source sono considerate più promettenti rispetto ad altre alternative proprietarie in quanto è ragionevole pensare che la disponibilità del codice sorgente garantisca una più veloce integrazione delle nuove soluzioni di IoT in tutti i domini applicativi.

Tabella 1- Piattaforme in esame

Ref	Platforms	a) Support of heterogeneous devices	b) Type	c) Architecture	d) Open source	e) REST	f) Data access control	g) Service discovery
1	AirVantage™	Needs gateway	M2M PaaS	Cloud-based	Libraries only (Apache v2, MIT and Eclipse v1.0	Yes	OAuth2	No
2	Arkessa	Yes	M2M PaaS	Cloud-based	No	n.a.	Facebook like privacy settings	No
3	ARM mbed	Embedded devices	M2M PaaS	Centralized/ Cloud-based	No	CoAP	User's choice	No
4	Carriots®	Yes	PaaS	Cloud-based	No	Yes	Secured access	No
5	DeviceCloud	Yes	PaaS	Cloud-based	No	Yes	n.a.	No
7	EveryAware	Yes	Server	Centralized	No	Yes	4 levels	No
8	Everyware	Needs gateway	PaaS	Cloud-based	No	Yes	n.a.	No
9	EvryThng	Yes	M2M SaaS	Centralized	No	Yes	Fine-grained	No
10	Exosite	Yes	PaaS	Cloud-based	Libraries only (BSD license)	Yes	n.a.	No
11	Fosstrack	RFID	Server	Centralized	No	No	Locally stored	No
12	GroveStreams	No	PaaS	Cloud-based	No	Yes	Role-based	No
13	H.A.T.	Home devices	PaaS	Decentralized	Yes	Yes	Locally stored	Yes
14	IoT-framework	Yes	Server	Centralized	Apache license 2.0	Yes	Locally stored	Yes
15	IFTTT	Yes	SaaS	Centralized	No	No	No storage	Limited
16	Kahvihub	Yes	Server	Centralized	Apache license 2.0	Yes	Locally stored	Yes
17	LinkSmart™	Embedded devices	P2P	Decentralized	LGPLv3	No	Locally stored	Yes
18	MyRobots	Robots	Robots PaaS	Cloud-based	No	Yes	2 levels	No
19	Niagara^{AX}	Yes	M2M SaaS	Distributed	No	n.a.	n.a.	n.a.
20	Nimbits	Yes	Server	Centralized/ Cloud-based	Apache license 2.0	Yes	3 levels	No
21	NinjaPlatform	Needs gateway	PaaS	Cloud-based	Open source hardware and Operating System	Yes	OAuth2	No
22	Node-RED	Yes	Server	Centralized	Apache license 2.0	No	User-based privileges	No
23	OpenIoT	Yes	Hub	Decentralized	LGPLv3	No	User-based privileges	Yes
24	OpenMTC	Yes	M2M client/ Server	Centralized/ Cloud-based	No	Yes	Secured access	No
25	OpenRemote	Home devices	Server	Centralized	Affero GNU Public License	Yes	Locally stored	No
26	Open.Sen.se	Ethernet enabled	PaaS/SaaS	Cloud-based	No	Yes	2 levels	Limited
27	realTime.io	Needs gateway	PaaS	Cloud-based	No	Yes	Secured access	No
28	SensorCloud™	No	PaaS	Cloud-based	No	Yes	n.a.	No
29	SkySpark	No	SaaS	Centralized/ Cloud-based	No	Yes	n.a.	No
30	Swarm	Yes	PaaS	Cloud-based	Client is open source (unknown license)	Yes	n.a.	n.a.
31	TempoDB	No	PaaS	Cloud-based	No	Yes	Secured access	No
32	TerraSwarm	Yes	OS	Decentralized	n.a.	n.a.	n.a.	Yes
33	The thing system	Home devices	Server	Centralized	M.I.T.	Yes	User's choice	No
34	Thing Broker	Yes	Server	Centralized	Yes	Yes	Locally stored	No
35	ThingSpeak	Yes	Server	Centralized/ Cloud-based	GNU GPLv3	Yes	2 levels	Limited
36	ThingSquare	Embedded devices	Mesh	Cloud-based	Gateway firmware is open source	Yes	No	No
37	ThingWorx	Yes	M2M PaaS	Cloud-based	No	Yes	User-based privileges	Yes
38	WoTkit	Yes	PaaS	Cloud-based	No	Yes	Secured access	Yes
39	Xively	Yes	PaaS	Cloud-based	Libraries are open source (BSD 3-clause), platform is not	Yes	Secured access	Yes

1.2.1 Gap Analysis

Nella sezione precedente abbiamo analizzato le caratteristiche delle piattaforme IoT che sono state individuate come importanti per lo sviluppo delle applicazioni di progetto. Tuttavia, molteplici sono ancora le lacune che possono essere identificate nella funzionalità offerta da queste piattaforme. Pertanto, si proporrà in questa sezione una gap analysis, che mira a valutare la maturità delle soluzioni attuali focalizzando l'analisi su più dimensioni.

Le dimensioni dell'analisi comprendono:

(i) L'estensibilità della piattaforma, ovvero la possibilità di supportare più tecnologie eterogenee di rilevamento e di attuazione.

(ii) La proprietà dei dati e le sue implicazioni per la sicurezza e la privacy.

(iii) L'elaborazione e la condivisione dei dati per il sostegno di nuovi servizi.

(iv) Il supporto degli sviluppatori alla piattaforma.

(v) L'integrità di un ecosistema IoT.

(vi) Marketplace di applicazioni e servizi di IoT.

L'estensibilità della piattaforma

L'essenza di una piattaforma IoT è quella di consentire la connessione sicura di una moltitudine sensori e attuatori eterogenei, aventi vincoli e capacità differenti, ad Internet. In assenza di uno standard di comunicazione univoco, sensori e attuatori di diversi fornitori possono sottoscrivere diversi modelli di interazione e possono implementare diversi protocolli di comunicazione tra quelli disponibili.

È ovvio quindi che il valore di una piattaforma IoT cresce proporzionalmente con il numero e la versatilità dei dispositivi supportati. Una piattaforma ideale IoT offrirà una serie di protocolli di comunicazione standardizzati in cui il produttore del dispositivo può selezionare gli opportuni protocolli (ad esempio, CoAP, HTTP, MQTT).

Nel caso di dispositivi passivi (ad es. attivati con RFID) o di dispositivi vincolati, la connettività si basa su un gateway di mediazione che deve essere completamente controllato dall'utente della piattaforma, simile a NinjaBlock [Piattaforma 21] il quale propone hardware e firmware open-source per il gateway.

Per favorire l'integrazione con dispositivi di rilevamento e attuazione, è essenziale che le comunità IoT adotti protocolli standardizzati validi per tutti i dispositivi, come è attualmente fatto per i dispositivi altamente vincolati da IETF o per le comunicazioni M2M da IEEE1888, ETSI M2M e 3GPP.

Attualmente i protocolli per i dispositivi vincolati sono supportati da OpenRemote [Piattaforma 25] (KNX, Z-Wave, X10 ecc.), LinkSmart TM (ZigBee), ARM mbed [Piattaforma 3] (MQTT, CoAP) e ThingWorx (MQTT); le altre piattaforme invece assumono l'uso di dispositivi relativamente potenti in grado di supportare protocolli

web tradizionali come nel caso di SensorCloud TM [Piattaforma 28], SkySpark [Piattaforma 29] o TempoDB [Piattaforma 31].

Infine, IFTTT [piattaforma 15], il quale comunica sia con i produttori dei dispositivi che con i fornitori di servizi web, propone soluzioni orientate alle esigenze di mercato (ad esempio, alle esigenze di Belkin) per estendere la piattaforma alle nuove tecnologie.

Come è possibile dedurre dall'analisi soprastante, soluzioni attuali dell'IoT si occupano di affrontare in modo diverso le interfacce dei dispositivi eterogenei. In generale, l'interoperabilità con i dispositivi è garantita sia mediante l'implementazione di un gateway che può essere esteso, ad esempio con l'aiuto di plug-in, per supportare nuovi tipi di dispositivi, oppure in grado di utilizzare un set limitato di protocolli supportati.

Come esempio, la piattaforma realTime.io [Piattaforma 27] ha proposto una connessione di sensori tramite un gateway proprietario (ioBridge che richiede anche l'uso di un protocollo di trasporto proprietario, ioDP), mentre la Thing-Worx [Piattaforma 37] e OpenMTC [Piattaforma 24] utilizzano Web sockets, MQTT o altri protocolli di comunicazione standard per collegare i dispositivi alla piattaforma. Altre piattaforme, come il The Thing System [Piattaforma 33] o H.A.T. mirano all'integrazione dei dispositivi presenti in ambienti "smart homes" e "smart places". Altre piattaforme, come Fosstrack [Piattaforma 11], consentono solo un tipo di tecnologia che, nel caso di Fosstrack, è RFID.

Si noti comunque che l'eterogeneità dei dispositivi supportati è limitata o è necessaria l'uso di un gateway, pertanto, per semplificare l'integrazione di nuovi tipi di dispositivi, i modelli di oggetti standard per i dispositivi IoT, dovrebbero essere ampiamente integrati dalle piattaforme IoT. Inoltre, i meccanismi di sicurezza, come nel dovrebbero essere integrati sulle piattaforme IoT per garantire una gestione sicura dei dispositivi IoT.

Stato dell'arte:
• Le piattaforme accettano solo oggetti intelligenti per parlare HTTP o richiedono gateway.
Aspettative:
• I dispositivi dovrebbero essere facilmente integrabili e integrabili con la piattaforma IoT. • Unificazione delle Risorse e semplificazione nell' usabilità.
Gaps:
• Supporto di dispositivi vincolati. • Modelli di dispositivi IoT standardizzati. • Autenticazione sicura, identificazione e gestione di dispositivi IoT.
Problemi:
• Interazioni eterogenee. • Standardizzazione del protocollo.

Raccomandazioni:
• Utilizzo di protocolli standard (ad esempio CoAP, LwM2M, MQTT). • Integrazione di protocolli di sicurezza e privacy di ultima generazione.

Proprietà dei dati

L'enorme volume di dati generato dai dispositivi nel mondo dell'Internet of Things impone una riflessione seria sulla la gestione dei dati soprattutto per quanto riguarda la necessità di mantenere un elevato grado di privacy e sicurezza.

Sulla base delle informazioni raccolte durante l'analisi delle piattaforme IoT di riferimento, la proprietà dei dati è il punto debole che interessa la maggior parte delle piattaforme. Alcune piattaforme cloud-based (ad esempio, Swarm [Platform 30]) assicurano che proprietà dei dati raccolti e memorizzati rimangano dei clienti. Tuttavia, la piena proprietà dei dati è raramente garantita. Nella maggior parte dei casi, i dati vengono inviati alla piattaforma in un formato crudo, memorizzati senza crittografia e si deducono ben poche informazioni sulle misure di sicurezza adottate per garantire la privacy.

Buona parte delle piattaforme elencate richiede l'utilizzo di chiavi di accesso o di altri meccanismi di controllo di accesso per ottenere le autorizzazioni di lettura o scrittura. I diritti di accesso sono determinati dagli utenti finali dei dispositivi, tramite un'interfaccia web (ThingSpeak [Piattaforma 35], Nimbits [Piattaforma 20]) o vengono lasciati gestire al provider di applicazioni (OpenRemote, Swarm, LinkSmart TM, Thing Broker [piattaforma 34]). Inoltre, la piattaforma EvryAware offre l'accesso ai feed di dati pubblici anche agli utenti anonimi che non richiedono chiavi di accesso. Tali impostazioni di privacy eccessivamente rigorose o troppo flessibili non forniscono un sufficiente controllo sui dati. Solo soluzioni, dove i dati vengono memorizzati localmente (ad esempio, H.A.T. o OpenRemote), offrono veramente la piena proprietà dei dati all'utente finale.

Ne deriva che le future soluzioni IoT debbano avere algoritmi e meccanismi che consentano al proprietario dei dati l'accesso solo a un insieme predefinito delle risorse e che i dati grezzi rimangano sotto il controllo dell'utilizzatore finale. Ad esempio, se il proprietario dei dati è disposto a archiviare i dati utilizzando un servizio offerto da un PaaS, questo deve essere in grado di crittografare i dati o di elaborarli prima di inviarli in cloud. Inoltre, poiché le impostazioni di privacy troppo rigide o troppo flessibili non forniscono un sufficiente controllo sui dati, nelle future soluzioni IoT, la visibilità dei dati deve essere accoppiata alla funzionalità di memorizzazione locale.

Stato dell'arte:

• Principalmente la proprietà è data all'utente finale, ma con politiche di privacy molto semplici.
Aspettative:
• Controllo completo dovrebbe appartenere al proprietario dei dati. • Memorizzazione in locale.
Gaps:
• Manipolazione di dati nei dispositivi di bordo (dispositivi più esterni al sistema). • Self-Storage.
Problemi:
• Sicurezza nell'archiviazione del dato.
Raccomandazioni:
• Sviluppo di algoritmi e meccanismi che consentano al proprietario dei dati l'accesso solo a un insieme predefinito delle risorse.

Elaborazione e condivisione dei dati

In generale, le applicazioni IoT possono essere caratterizzati da grandi volumi di dati ed spesso possono disporre del requisito del Real-Time. Pertanto questi dati potrebbero essere spesso inaffidabili, incompleti e avere livelli qualità differenti. Se consideriamo che questi dati sono anche rappresentati in diversi formati e da modelli diversi la complessità del problema aumenta. Ad esempio, è quasi impossibile utilizzare direttamente il dato di basso livello fornito da un sensore senza un modello di conoscenza ben definito.
Pertanto se ne deduce che dati e le conoscenze dietro i dati sono il nucleo della ricchezza prodotta dall' IoT.
Le soluzioni IoT ad oggi sviluppate non supportano, o hanno limitato il supporto per la elaborazione e la condivisione dei flussi di dati. Tuttavia, è possibile combinare flussi multipli in una singola applicazione se si conosce l' URI alle fonti di informazioni desiderate, ma ciò rappresenta una sfida tecnica per gli sviluppatori di applicazioni.
L' IoT-Framework di Ericsson [piattaforma 14] fornisce algoritmi e meccanismi in grado di integrare flussi virtuali che combinati con flussi locali permettono la visualizzazione dei dati, le analisi statistiche e le predizioni.
C'è inoltre da sottolineare che altre tecniche di elaborazione dati sono state proposte ma tuttavia la loro aggregazione nelle piattaforme IoT è ancora limitata.

Stato dell'arte:
• Formato di condivisione dati non uniforme. • La condivisione viene eseguita tramite una REST API non uniforme .

Aspettative:
• Formato di dati uniforme su più piattaforme. • Meccanismo Pub / Sub e data catalog.
Gaps:
• L'elaborazione dei dati non è ben integrata nelle piattaforme IoT. • Analisi dei dati è disponibile solo in soluzioni cloud-based. • Mancanza di data catalogs.
Problemi:
• Mancanza di una fusion efficace di flussi di dati da più data catalogs. • I dispositivi IoT hanno capacità di calcolo limitate.
Raccomandazioni:
• Data catalogs con indici semantici. • Modelli dati uniformi e interoperabili. • Integrazione delle tecnologie di elaborazione dati nelle piattaforme.

Il supporto degli sviluppatori alla piattaforma.

Per favorire uno sviluppo accelerato delle applicazioni, le piattaforme IoT dovrebbero fornire agli sviluppatori delle interfacce di programmazione (API) flessibili, preferibilmente con l'ausilio di primitive al livello di astrazione più elevato. Inoltre, per consentire uno sviluppo efficiente delle applicazioni per piattaforme cross-IoT, queste API devono essere uniformi tra le piattaforme il più possibile.

Le piattaforme IoT di oggi forniscono quasi sempre un'API pubblica per accedere ai servizi. Le API sono di solito basate su principi RESTful e consentono operazioni comuni come PUT, GET, PUSH o DELETE.

Queste operazioni permettono l'interazione con i dispositivi collegati alla piattaforma e la loro gestione.

Solo quattro delle piattaforme studiate non hanno incluso un'API REST per facilitare lo sviluppo di servizi web (ad esempio Fosstrack, LinkSmart TM, IFTTT e OpenIoT). Le altre piattaforme, tuttavia, utilizzano API e modelli di dati non uniformi di 3 REST che complicano l'analisi dei dati su più piattaforme.

Molte piattaforme offrono anche librerie, in alcuni casi open source (ad esempio AirVantage TM [Platform 1], Exosite [Piattaforma 10], IoT-Framework o Xively), che permettono di fare associazioni alle REST API anche per differenti linguaggi di programmazione. Tuttavia, queste librerie di associazioni comprendono solo funzionalità di base e pertanto non garantiscono un valido supporto agli sviluppatori di applicazioni.

In una certa misura, alcune piattaforme come ThingSpeak consentono di creare widget scritti, in Javascript, HTML e CSS, che possono essere distribuiti ad altri utenti tramite la piattaforma. In alternativa, la Carriots

platform [piattaforma 4] fornisce un completo kit di sviluppo software (SDK), scritto in Groovy, a supporto degli sviluppatori di applicazioni.

Un definitivo passo avanti sarebbe quello di sviluppare un approccio più generalizzato nelle soluzioni IoT per massimizzare l'usabilità dei servizi forniti dalle piattaforme IoT. Oltre alle API, potrebbe essere definito un linguaggio specifico del dominio (DSL) per semplificare lo sviluppo delle applicazioni IoT, anche offrendo primitive funzionali che descrivono lo spazio a un livello di astrazione più elevato.

Stato dell'arte:
• Utilizzo di REST API per accedere ai dati o ai dispositivi gestiti dalla piattaforma • Le applicazioni sono per uso interno piuttosto che per la condivisione (ad eccezione di IFTTT)
Aspettative:
• Sviluppo di API comuni per facilitare lo sviluppo di applicazioni cross-platform. • Domain Specific Language (DSL) dedicato allo sviluppo cross-platform delle applicazioni.
Gaps:
• Presenza limitata di SDK. • Assenza di DSL con primitive ai più alti livelli di astrazione.
Problemi:
• Standardizzazione delle interazioni applicative dedicate allo IoT. • Assenza di gli App store dedicate all' IoT.
Raccomandazioni:
• Le piattaforme IoT devono fornire SDK e API che ottimizzano la riusabilità dei servizi forniti dalla loro piattaforma.

L'integrità di un ecosistema IoT

Dalle analisi svolte si deduce chiaramente come il successo di una piattaforma IoT dipenda dall'esistenza di un ecosistema ben strutturato dove gli acquirenti, i fornitori e i produttori di dispositivi o servizi forniscono collettivamente una varietà di applicazioni, prodotti, e servizi agli utenti finali di IoT.

In tal senso la piattaforma IoT definisce il nucleo del suo ecosistema fornendo a tutti i membri un serie comune di risorse condivise essenziali per i loro prodotti e servizi.

Inoltre, per prosperare, la piattaforma dovrebbe essere facilmente espandibile dagli sviluppatori e complementarmente dovrebbe offrire loro incentivi per innovare e contribuire alla sviluppo della piattaforma stessa. In altre parole, la piattaforma dovrebbe attirare a se gli sviluppatori di componenti aggiuntivi e di

applicazioni, ma tuttavia, solo le piattaforme open source riescono a rispondere efficacemente a questa sfida garantendo tempestività e scalabilità per affrontare l'emergere di nuove tecnologie.
Al contrario le piattaforme proprietarie non consentono di aggiungere componenti aggiuntivi, ad eccezione di IFTTT e ThingWorx i quali fanno uso di strumenti di terze parti.
Una soluzione interessante per consentire di trattare i domini di IoT come un unico ecosistema convergente che fornisce prodotti e servizi innovativi e consente un'economia di scala, potrebbe essere quella di utilizzare un mediatore di piattaforme IoT con il compito di agevolare la condivisione di applicazioni e servizi anche in tutti gli altri sotto-ecosistemi di IoT. Tuttavia, la possibilità di intermediazione multipiattaforma non è stata approfondita e gli ecosistemi di IoT in esame rappresenta una moltitudine di silos verticali in frammentazione.

Stato dell'arte:
• Le piattaforme forniscono tools, storage e run-time di ambiente agli sviluppatori di applicazioni.
Aspettative:
• Sviluppo di piattaforme facilmente espandibili dagli sviluppatori. • Condivisione di applicazioni e servizi cross-platform.
Gaps:
• Limitata espandibilità delle piattaforme. • Possibilità di monetizzazione limitate. • Supporto limitato per l'integrazione cross-platform.
Problemi:
• Soluzioni platform-specific. • L'utilizzatore delle multi-piattaforme riscontra difficoltà nell'integrazione.
Raccomandazioni:
• È necessario un broker per facilitare l'integrazione tra piattaforme • Sviluppo di specifici modelli che aiutino a definire contestualmente le applicazioni IoT allo scopo di essere conosciute e utilizzate da parte degli utenti finali.

Marketplace di applicazioni e servizi di IoT

I Marketplace di applicazioni software sono finalizzati a facilitare la scoperta, l'acquisto e la distribuzione delle applicazioni. Questi mercati possono essere esemplificati con soluzioni hardware-specific e controllate centralmente, quali Apple App Store o Google Play o mercati hardware-agnostic come Good, Handster, Nexva e

SlideMe. La disponibilità e l'accessibilità a tali mercati è fondamentale per la diffusione delle innovazioni software in generale, e in particolare di innovazioni IoT. Tuttavia, i tradizionali negozi di App sembrano avere restrittive limitazioni per quanto riguarda il mondo IoT.

Tra le piattaforme IoT in analisi solo alcune dispongono di negozi dedicati per le applicazioni (ad es. ThingWorx), e ancor meno (IFTTT) permettono di condividere pubblicamente le applicazioni o (OpenIoT) permettono di far pagare, in base all'effettivo utilizzo, le applicazioni agli utenti finali.

Inoltre, una delle sfide chiave di IoT è sfruttare sapientemente tutti i dati attualmente in commercio. Le aziende già raccolgono enormi quantità di dati derivanti da sensori, ma tali dati vengono utilizzati solo per rilevare e controllare le anomalie. Se questi dati venissero utilizzati anche altri fini, come l'ottimizzazione del servizio e la previsione, che forniscono il vero valore aggiunto in un applicazione, si avrà la necessità di sviluppare nuove tipologie mercati basati su dati IoT attraverso i quali prospereranno nuove interazioni commerciali (cioè, business-to-business). La piattaforma Windows Azure Data Market fornisce già un esempio di modello di business di successo. Ad esempio, la piattaforma consente alle aziende di pubblicare grandi flussi di dati e renderli disponibili a un gran numero di sviluppatori. La piattaforma attraverso specifici abbonamenti offre pertanto l'accesso a tali dati. Si ritiene che lo sviluppo di piattaforme dedicate a IoT, simile al Windows Azure Data Market, sia un requisito essenziale per la sostenibilità delle soluzioni IoT. La *Figura 9* illustra le opportunità che emergono dalla disponibilità di mercati dedicati all'IoT.

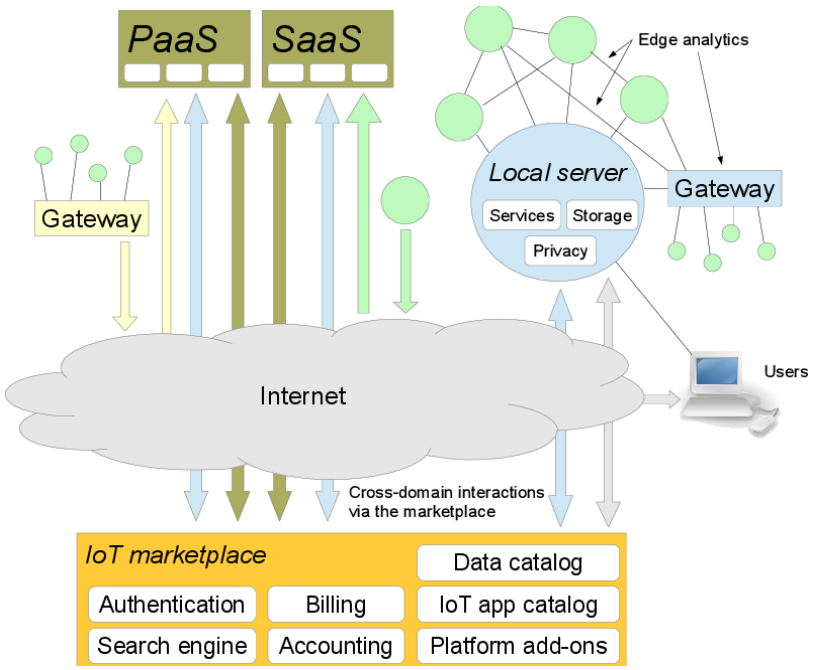

Figura 9

Stato dell'arte:
• Condivisione limitata di applicazioni • Alcune piattaforme propongono applicazioni che hanno un costo basato sull' loro effettivo utilizzo da parte degli utenti finali
Aspettative:
• Data catalogs IoT dedicati, app store IoT e marketplaces di dispositivi IoT • Pubblicizzare, consegnare e addebitare l'utilizzo di applicazioni e dati
Gaps:
• Risultano mancanti dati, applicazioni e dispositivi dedicati specificatamente all' IoT. • Generalmente manca la fatturazione degli utenti finali dei dati
Problemi:
• Assenza di ecosistema di sviluppatori indipendenti, produttori di dispositivi e utenti finali che supportano la piattaforme per rendere sostenibile la domanda di mercato
Raccomandazioni:
• Il marketplace delle funzionalità dovrà essere fornito dalle future piattaforme IoT

Analisi delle soluzioni esistenti per le funzionalità di data management, access e analytics

2.1 Funzionalità di data management

Un sistema di gestione di basi di dati (in inglese Data Base Management System, DBMS) è un sistema software in grado di gestire collezioni di dati che siano *grandi*, *condivise* e *persistenti*, assicurando la loro *affidabilità* e *privatezza*. Come ogni prodotto informatico, un DBMS deve essere *efficiente* ed *efficace*. Una base di dati è una collezione di dati gestita da un DBMS. Precisiamo di seguito le caratteristiche dei DBMS e delle basi di dati:

- Le basi di dati sono ***grandi*** nel senso che possono avere anche dimensioni enormi e comunque in generale molto maggiori della memoria centrale disponibile. Alla data di redazione del presente elaborato, le più grandi basi di dati hanno dimensioni dell'ordine delle centinaia di terabyte e contengono migliaia di record. Ovviamente possono esistere anche basi di dati "piccole", ma i sistemi devono poter gestire i dati senza porre limiti alle dimensioni, a parte quelle fisiche dei dispositivi.

- Le basi di dati sono ***condivise***, nel senso che applicazioni e utenti diversi devono poter accedere, secondo opportune modalità, a dati comuni. È importante notare che in questo modo si riduce la *ridondanza* dei dati, poiché si evitano ripetizioni, e conseguentemente si riduce anche la possibilità di *inconsistenze:* se esistono varie copie degli stessi dati, è possibile che esse, in qualche momento, non siano uguali. Per garantire l'accesso condiviso ai dati da parte di molti utenti che operano contemporaneamente, il DBMS dispone di un meccanismo apposito, detto *controllo di concorrenza*.
- Le basi di dati sono ***persistenti,*** cioè hanno un tempo di vita che non è limitato a quello delle singole esecuzioni dei programmi che le utilizzano.
- I DBMS garantiscono la ***affidabilità***, cioè la capacità del sistema di conservare sostanzialmente intatto il contenuto della base di dati (o almeno permetterne la ricostruzione) in caso di malfunzionamenti e software. A questo scopo i DBMS forniscono specifiche funzionalità di *salvataggio* e *ripristino* (backup e recovery).
- I DBMS garantiscono la ***privatezza*** dei dati. Ciascun utente, riconosciuto in base a un nome d'utente che è specificato all'atto di interagire con il DBMS, viene abilitato a svolgere solo determinate azioni sui dati, attraverso meccanismi di *autorizzazione*.

Modello dei dati

Un modello dei dati è un insieme di concetti utilizzati per organizzare i dati di interesse e descriverne la struttura in modo che essa risulti comprensibile a un elaboratore. Il *modello relazionale* dei dati (SQL), attualmente il più diffuso, permette di definire tipi per mezzo del costruttore *relazione*, che consente di organizzare i dati in insieme di record a struttura fissa. Oltre al modello relazionale sono stati definiti altri tre tipi di modelli:

- Il *modello gerarchico*, basato sull'uso di strutture ad albero, definito durante la prima fase di sviluppo dei DBMS, ma tuttora ampiamente utilizzato;
- Il *modello reticolare*, basato sull'uso di grafi, sviluppato successivamente al modello gerarchico;
- Il *modello ad oggetti*, sviluppato come evoluzione del modello relazionale, che estende alle basi di dati il paradigma di programmazione a oggetti.

I modelli dei dati precedentemente elencanti sono effettivamente disponibili su DBMS commerciali; essi vengono detti *logici* per sottolineare che le strutture utilizzate da questi modelli, pur essendo astratte, riflettono una particolare organizzazione. Più recentemente, sono stati introdotti altri modelli, detti *concettuali*, utilizzati per descrivere i dati in maniera completamente indipendente dalla scelta del modello logico. Il loro nome deriva dal fatto che essi tendono a descrivere i *concetti* del mondo reale, piuttosto che i dati utili a rappresentarli.
I database *non relazionali* sono nettamente differenti da quelli relazionali. I dati sono conservati in documenti, non in tabelle, per cui la prima differenza con i DB relazionali sta nel fatto che le informazioni non sono distribuite in differenti strutture logiche, ma vengono aggregate per oggetto in documenti la cui natura può essere di tipo Key-Value (che rappresenta la forma primitiva di database NoSQL) o Document Store basati

su semantica JSON. Ogni documento aggregato raccoglie tutti i dati associati a un'entità, in modo che qualsiasi applicazione possa trattare l'entità come oggetto e valutare in un sol colpo tutte le informazioni a essa correlate. In questo modo, si evitano anche i fardelli computazionali dovuti ai passaggi di aggregazione delle informazioni tipici del linguaggio SQL, in quanto tutti i dati necessari e corrispondenti a un medesimo oggetto sono già disponibili in un unico documento.
L'assenza di tabelle permette ai database non relazioni di essere schemaless, ossia privi di un qualsiasi schema definito a priori e questa caratteristica conferisce ai DB NoSQL un altro vantaggio non trascurabile.

Vantaggi e svantaggi dei DBMS
Dalle caratteristiche citate si intravedono già i vantaggi di una soluzione NoSQL rispetto a una SQL, che possiamo così esplicitare.

1. **Leggerezza computazionale**: i database NoSQL non prevedono operazioni di aggregazione sui dati, in quanto tutte le informazioni sono già raccolte in un unico documento associato all'oggetto da trattare. Negli ambienti SQL la complessità di queste operazioni, e quindi il peso computazionale, cresce con l'ingigantirsi della base di dati, del numero di tabelle e delle informazioni da trattare. Il NoSQL, invece, non ha limiti di dimensioni in questo senso. Così si ottengono migliori prestazioni e performance anche in ambienti di Big Data. Lo scotto da pagare a tutta questa flessibilità e alla proprietà di aggregazione dei database NoSQL è la **duplicazione delle informazioni**. In realtà, i costi sempre meno proibitivi dei sistemi di storage rendono questo svantaggio poco importante.

2. **Assenza di schema**: i database NoSQL sono privi di schema in quanto il documento JSON contiene tutti i campi necessari, senza necessità di definizione. In questo modo, possiamo arricchire le nostre applicazioni di nuovi dati e informazioni, definibili liberamente all'interno dei documenti JSON **senza rischi per l'integrità dei dati**. I database non relazionali, a differenza di quelli SQL, si rivelano quindi adatti a inglobare velocemente nuovi tipi di dati e a conservare dati semistrutturati o non strutturati.

3. **Scalabilità orizzontale garantita**: l'aggregazione dei dati e l'assenza di uno schema definito a priori offre l'opportunità di scalare orizzontalmente i database NoSQL senza difficoltà e senza rischi operativi.

2.2 Funzionalità di data access

L'accesso ai dati si riferisce al software e alle attività relative alla memorizzazione e al recupero di dati alloggiati all'interno del repository. Gli utenti che hanno accesso ai dati possono **memorizzare, recuperare, spostare o manipolare** dati memorizzati, su una vasta gamma di unità disco rigido e dispositivi esterni.
Esistono due modi per accedere ai dati memorizzati: **accesso casuale e accesso sequenziale.** Il metodo sequenziale richiede che le informazioni siano spostate all'interno del disco utilizzando un'operazione di ricerca

fino a quando non si trovano i dati. Ogni segmento di dati deve essere letto uno dopo l'altro fino a quando non si trovano i dati richiesti. La lettura dei dati consente agli utenti di memorizzare o recuperare i dati in qualsiasi punto del disco e di accedere ai dati in tempo costante.
Mentre quando si utilizza l'accesso casuale, i dati sono suddivisi in più parti o pezzi e si trovano ovunque in modo casuale su un disco. I file sequenziali sono di solito più veloci da caricare e recuperare perché richiedono meno operazioni di ricerca.
Un esempio di tecnologia di data access sviluppata da Microsoft.NET framework è ADO.NET che fornisce la comunicazione tra sistemi relazionali e non relazionali attraverso un insieme comune di componenti.

2.3 Funzionalità di data analytics

L'analisi dei dati (DA, data analytics) è il processo che, con l'aiuto di sistemi software specializzati, **esamina dei set di dati per trarre conclusioni circa le informazioni che contengono.** Le tecnologie e le tecniche di data analytics sono ampiamente utilizzate dalle industrie commerciali per prendere decisioni di business e dagli scienziati e ricercatori per formulare modelli, teorie e ipotesi scientifiche.
Come termine, il data analytics si riferisce prevalentemente ad un assortimento di applicazioni, che vanno dalla business intelligence di base (BI), al reporting e all'analisi analitica online (OLAP) fino a diverse forme di analisi avanzate.
Le iniziative di data analytics possono aiutare le aziende a incrementare i ricavi, migliorare l'efficienza operativa, ottimizzare le campagne di marketing e gli sforzi del servizio clienti, rispondere più rapidamente alle tendenze del mercato emergente e ottenere un vantaggio competitivo rispetto ai rivali - tutto con l'obiettivo finale di aumentare le prestazioni aziendali.
Ad alto livello, le metodologie di analisi dei dati includono l'analisi dei dati esplorativi (EDA, exploratory data analysis) che cerca di individuare modelli e relazioni nei dati e l'analisi dei dati di conferma (CDA, confirmatory data analysis), che applica tecniche statistiche per determinare se le ipotesi su un set di dati siano vere o false.
Le analisi dei dati possono anche essere classificate in analisi quantitative e analisi qualitativi. Nel primo caso viene eseguita un'analisi di dati numerici con variabili quantificabili che possono essere confrontate o misurate statisticamente. L'approccio qualitativo è più interpretativo: si concentra sulla comprensione del contenuto di dati non numerici come testo, immagini, audio e video, comprese frasi comuni, temi e punti di vista.

Tipologie di dato e policies di accesso, condivisione e management

3.1 Overview delle tipologie e dei formati di rappresentazione dei dati in piattaforme di condivisione

La comunicazione può essere definita come l'atto di trasferire le informazioni da una entità ad un'altra, rispettando specifiche regole reciprocamente comprese. Il mittente codifica le informazioni che vengono trasmesse e il ricevitore decodifica il messaggio per capirlo.
La forma dell'informazione codificata è in genere definita come il formato dei dati.
Il formato dei dati comprende anche le regole di codifica e decodifica dei dati, per evitare che sorgano ambiguità e quindi impedire un'erronea interpretazione dei dati (incomprensioni).
In questa sottosezione forniremo una panoramica di tipologie e di formati di rappresentazione dei dati tra le più utilizzate nelle più comuni piattaforme di condivisione.
In primo luogo, introduciamo due formati comuni di dati. Cominciamo con le tecnologie generiche per serializzare i dati arbitrari e quindi procedere agli approcci che vengono utilizzati principalmente per serializzare i dati semantici basati sul modello di descrizione delle risorse (RDF).

Extensible Markup Language (XML) è un formato dati molto importante applicato in molti domini, ad esempio in diversi tipi di protocolli di comunicazione, per la definizione di Interfaccia Utente o in applicazioni grafiche. Uno dei punti forti di XML è lo schema di linguaggio che consente la definizione di un meta-modello per la memorizzazione dei dati. Ciò consente di modellare strutture dati in modo molto preciso e consente la convalida di istanze XML rispetto al rispettivo schema. Questo è il motivo principale per cui XML ha avuto molto successo negli ambienti industriali.

JavaScript Object Notation (JSON) è un semplice formato per lo scambio di dati, soprattutto nelle applicazioni client-server. E' un formato di testo completamente indipendente dal linguaggio di programmazione, anche se utilizza convenzioni che ricordano linguaggi della famiglia del C, come C, C++, C#, Java, JavaScript, Perl, Python, e molti altri.

The Efficient XML Interchange (EXI) è un formato di scambio binario ed è una rappresentazione molto compatta di XML che è stata sviluppata per ottimizzare le prestazioni e l'utilizzo di risorse computazionali. EXI basa l'ottimizzazione su un approccio guidato dalla grammatica, utilizzando un modulo chiamato EXI processor per codificare i dati XML negli streams EXI o per decodificare e rendere i dati nuovamente fruibili.
EXI è abbastanza comune negli oggetti intelligenti con risorse limitate come scarsa memoria, bassa potenza di calcolo e ridotto utilizzo della larghezza di banda. Tale tecnologia è arrivata alla versione 1.0 ed è entrata a far parte delle raccomandazioni W3C a marzo del 2011.

EXI4JSON è un formato rilasciato nel 2016 e mira ad essere uno standard riconosciuto da W3C.
JSON è molto popolare a causa di Java e JavaScript, tuttavia, non è adatto per l'utilizzo in dispositivi con risorse limitate. EXI4JSON è stato sviluppato per portare i dati basati su JSON in una rappresentazione binaria e renderli quindi utilizzabile nel mondo IoT per oggetti intelligenti con risorse limitate. EXI4JSON ha un TRL di 6-7

Resource Description Framework (RDF/XML) è un modello di descrizione astratto e non pone vincoli sulla sintassi e sul significato delle descrizioni di una risorsa. Questo vuol dire che ognuno potrebbe proporre un qualsiasi meccanismo per descrivere risorse utilizzando le astrazioni previste da RDF. Ad esempio, una descrizione RDF potrebbe essere espressa tramite una rappresentazione grafica delle risorse, degli attributi e delle asserzioni.
Tuttavia, per ragioni pratiche, la definizione formulata del W3C propone XML come metalinguaggio per la rappresentazione del modello RDF, cioè propone RDF come linguaggio basato su XML.

JSON-LD (linking data) è un formato di interscambio di Linked data, che utilizza JSON. Si può utilizzare su

qualsiasi piattaforma, indipendentemente dalla lingua di programmazione utilizzata e si usa per lo scambio e la memorizzazione di dati strutturati nelle applicazioni web e nelle app. Si tratta di una raccomandazione del Consorzio W3C, sviluppata in origine dal Gruppo JSON for Linking Data e trasferita in seguito all'RDF Working Group per revisioni, miglioramenti e standardizzazione.

Triple-based representation (Turtle) è una rappresentazione testuale di un grafico RDF che non segue una struttura ad albero (come ad esempio XML e JSON). È una delle varianti di serializzazione di RDF comunemente usate. Turtle è una norma di raccomandazione W3C e è supportata da molti strumenti basati sul web semantico (ad esempio, Protégé e Apache Jena).

Notation3, o N3, come è comunemente noto, è una serializzazione non stenografica XML di modelli Struttura Descrizione Resource, progettata con la leggibilità umana: N3 è molto più compatta e leggibile della notazione XML RDF.

N-Triples è un formato per la memorizzazione e la trasmissione dei dati. È un formato di serializzazione di testo semplice e lineare per i grafici RDF (Resource Description Framework) ed è un sottoinsieme del formato Turtle (Terse RDF Triple Language). N-Triples è stato progettato per essere un formato dati più semplice di Notation 3 e Turtle, e quindi più facile per il software da analizzare e generare. Tuttavia, poiché manca di alcune delle *shortcuts* fornite da altre serializzazioni RDF, può risultare difficile leggere o digitare manualmente grandi quantità di dati.

RDF/EXI si basa sul formato dei dati EXI. Formati tradizionali di serializzazione RDF, come RDF / XML, Turtle e JSON-LD si basano su una rappresentazione in testo semplice. Questo non facilmente supportabile in un ambiente IoT che si compone di dispositivi poveri di risorse come i microcontrollori con memoria limitata e scarsa capacità di elaborazione e larghezza di banda.
La forza RDF-EXI è la riduzione dei contenuti basati su stringhe, nonché la ridondanza che è fortemente utilizzata nei dati basati su RDF (in particolare URIs).

3.2 Analisi delle policies di accesso, condivisione e gestione dei dati nel Cloud Computing

Il Cloud Computing è una tecnologia applicabile a ogni settore (general purpose technology) che consente di sostituire hardware e software con collegamenti on–line a centri dati remoti, permettendo a tutti di poter ottenere

in modo scalabile capacità di calcolo e di memorizzazione, dati, informazioni, etc. compatibilmente con l'esigenza di garantirne la sicurezza e la riservatezza. Una componente molto importante e non trascurabile per i servizi Cloud è la sicurezza delle reti e delle informazioni, la protezione dei dati e la privacy. Tali aspetti rappresentano sicuramente una delle sfide più importanti per la gestione dei sistemi Cloud. Le policy di sicurezza vengono sviluppate per garantire la conformità delle organizzazioni a leggi e regolamenti, ma anche per proteggere le informazioni sui client.

Analisi dei rischi per la sicurezza e per la privacy nel Cloud

Il tipo di attività, la quantità di dati in outsourcing e il fornitore del servizio selezionato sono i fattori da cui dipendono i rischi associati al Cloud Computing. Infatti, la tecnologia Cloud presenta diverse problematiche, come quelle legate a data privacy, a data lock-in e a scalable storage, oltre alle problematiche di tipo legale, che sono strettamente legate ai fornitori e all'esigenza di avere infrastrutture in vari luoghi geografici. A tal proposito e come già detto in precedenza, gli utenti non sono a conoscenza della posizione fisica dei dati affidati al fornitore del servizio. Cosa che può sembrare irrilevante, ma è proprio il luogo dove si trovano i dati a determinare il tipo di normativa applicabile per la loro protezione. Altro aspetto che gli utenti di servizi Cloud devono considerare è l'informazione sulla gestione dei dati archiviati. I client devono conoscere le procedure adottate dal fornitore di servizio Cloud per quanto riguarda il recupero dei dati, sia nel caso di violazione della sicurezza sia nel caso della perdita dei dati. Inoltre, l'utente, affidando i dati ai sistemi di fornitori remoti, ne perde il controllo diretto ed esclusivo. Il fornitore si assume la responsabilità di preservare l'integrità, la riservatezza e la disponibilità dei dati, in base alla tipologia di servizi offerti all'utente.
Attualmente tali aspetti sono molto discussi e non esiste un'unica visione accettata nella comunità scientifica. Gartner [8] individua sette rischi per la sicurezza:

- privilegi di accesso per gli utenti;
- conformità alle normative;
- ubicazione dei dati;
- separazione dei dati;
- recupero dei dati;
- indagine degli errori;
- sostenibilità a lungo termine.

La Cloud Security Alliance, CSA [5], associazione internazionale no-profit, indica una raccolta di best practices per quattordici domini:

- ✓ architettura del Cloud;
- ✓ governance e gestione del rischio aziendale;
- ✓ aspetti legali su contratti e ricerca elettronica;

- ✓ conformità e audit;
- ✓ gestione delle informazioni e sicurezza dei dati;
- ✓ interoperabilità e portabilità;
- ✓ sicurezza tradizionale, continuità operativa e disaster recovery;
- ✓ operazioni di data center;
- ✓ risposta agli incidenti, notifica e risanamento;
- ✓ sicurezza applicativa; cifratura e gestione delle chiavi;
- ✓ gestione delle identità e degli accessi;
- ✓ virtualizzazione;
- ✓ security as a service.

I principali rischi per la sicurezza dei servizi Cloud indicati dalla Cloud Security Alliance – Italy Chapter [5] sono i seguenti:

- Perdita di governance. L'utente deve cedere il controllo al Cloud Provider su alcuni aspetti e i Service Level Agreement (SLA) possono non fornire alcuni servizi, lasciando delle lacune nella sicurezza;
- Lock-in. Il client ha spesso difficoltà a migrare da un fornitore di servizi Cloud a un altro, così come ha difficoltà nel riportare indietro dati e servizi verso un ambiente IT interno. In particolare, nel caso in cui non è abilitata, da parte del fornitore, la portabilità dei dati, delle applicazioni e dei servizi;
- Compromissione dell'interfaccia di gestione. Le interfacce di gestione per il client di un fornitore di servizi di un Cloud Pubblico sono accessibili tramite Internet e mediano l'accesso a una maggiore quantità di risorse, aumentando i rischi. In particolar modo quando sono combinati con la vulnerabilità dei browser web e con gli accessi remoti;
- Protezione dei dati. In questo caso, i rischi sono di diverso tipo sia per i clienti sia per i fornitori di servizi Cloud. L'utente può riscontrare, in alcuni casi, difficoltà nel verificare la gestione lecita dei dati da parte del cloud provider. Bisogna dire che alcuni fornitori di servizi Cloud danno informazioni sui metodi di gestione dei dati oppure forniscono sintesi delle certificazioni delle proprie attività e della gestione dei dati, come la certificazione SAS70;
- Cancellazione insicura o incompleta dei dati. Nel momento in cui viene fatta una richiesta di cancellazione di una risorsa nel Cloud, questa potrebbe non risolversi con una reale eliminazione dei dati. Un'eliminazione o una tempestiva eliminazione dei dati potrebbe essere impossibile, perché le copie extra dei dati sono immagazzinate ma non disponibili oppure perché il disco, contenente i dati da eliminare, contiene anche i dati di altri clienti. Nel caso di multi-tenancy e di riutilizzo delle risorse hardware, si hanno più rischi rispetto al caso di hardware dedicato.

In generale, l'adozione di un modello di Cloud Computing comporta l'esigenza di esaminare non solo gli aspetti

tradizionali di sicurezza, quali la gestione degli accessi e la protezione dei dati, ma anche aspetti specifici, come la corretta definizione di trattamento dei dati e la separazione tra la titolarità e la responsabilità dei dati, che, se non considerati in maniera appropriata, potrebbero causare rischi per la confidenzialità, l'integrità e la disponibilità delle informazioni.

Titolarità delle informazioni

Il termine trattamento dei dati indica qualunque operazione o insieme di operazioni per la raccolta, la registrazione, la gestione, la diffusione e altre azioni sui dati personali. In particolare, si individuano quattro attori principali:

1. L'interessato, ovvero la persona, l'ente o l'associazione a cui si riferiscono i dati personali;
2. Il titolare, ovvero la persona, l'ente o la Pubblica Amministrazione a cui compete la modalità del trattamento dei dati personali;
3. Il responsabile, ovvero la persona, la Pubblica Amministrazione o qualsiasi altro ente che è responsabile della protezione e della gestione dei dati personali;
4. L'incaricato, ovvero la persona fisica delegata dal titolare o dal responsabile a compiere azioni di trattamento dei dati.

Stabilire la titolarità dei dati è un punto fondamentale per la loro protezione da possibili manipolazioni non autorizzate.
Con riferimento ai dati che costituiscono un documento, è possibile identificare differenti ruoli per la definizione della titolarità:

- creatore: l'utente o l'entità responsabile della creazione del documento;
- autore: l'utente o l'entità responsabile del contenuto del documento;
- responsabile: l'utente o l'entità responsabile della protezione e della gestione del documento.

In generale, il titolare dei dati coincide con la persona che li ha creati oppure con l'ente a cui il creatore afferisce. Eventuali aggiornamenti e/o cancellazioni di informazioni devono avvenire solo se l'utente, che intende realizzare tali operazioni, è in possesso della titolarità sui dati da alterare.
Ad oggi, il quadro giuridico non specifica le responsabilità relative al trattamento dei dati in un "sistema di erogazione dei servizi" di tipo Cloud. Tali limitazioni e carenze a livello giuridico possono essere superate prevedendo due possibili scenari: il primo scenario consiste nell'assegnare al fornitore di servizi Cloud il ruolo di co–titolare del trattamento; nel secondo scenario, più realisticamente, il fornitore di servizi Cloud assume il ruolo di responsabile del trattamento. Risulta, pertanto, di fondamentale importanza definire opportune clausole contrattuali che tutelino l'utente di servizi Cloud.
Esistono differenti tecniche per proteggere la titolarità dei dati. Ad esempio, è possibile ricorrere a una

combinazione di tecniche di cifratura o di watermarking (letteralmente “filigranatura”), che possono essere applicate ai dati contenuti in un documento. Nel primo caso, il documento contiene il risultato dell’attuazione di un apposito algoritmo di cifratura sul suo contenuto in chiaro. Nel secondo caso, si effettua l’inclusione di informazioni all’interno del documento da proteggere. Tali informazioni aggiuntive devono rendere manifesto a tutti gli utenti chi sia il proprietario del documento. Uno dei problemi della cifratura è la gestione delle chiavi, ovvero dove memorizzare le chiavi e come proteggerle quando le si usano, al fine di evitare che siano violate o rubate. La soluzione allo scambio e alla protezione delle chiavi può essere a carico delle applicazioni, supportata dalla piattaforma di Cloud scelta oppure da strumenti ad hoc per la sicurezza nel Cloud, come nel caso di Porticor® Virtual Private Data (VPD™). Anche le tecniche di watermarking possono essere soggette a sofisticati tipi di attacchi volti a violarne le garanzie di sicurezza.

Autenticità e autenticazione

Ogni utente, che effettua operazioni sui dati di un documento, deve prendersi la responsabilità delle proprie azioni, anche da un punto di vista legale. Pertanto, un utente che intende accedere alle funzionalità o ai dati di un’applicazione deve esibire un insieme di affermazioni, dette security claims, che consentono di stabilire l’autenticità delle sue azioni. Tali security claims devono essere opportunamente certificati, al fine di verificare che l’utente è effettivamente chi dichiara di essere (autenticazione). Un esempio di queste affermazioni possono essere il nome del soggetto, il suo ruolo, l’associazione con altri soggetti, le preferenze e/o le capacità. Esse sono tipicamente impacchettate in un artefatto chiamato security token, il quale può essere realizzato dal soggetto stesso e certificato, per esempio, attraverso l’uso di una smartcard, oppure emesso da un’altra entità.
All’atto della richiesta dei servizi e dati, bisogna esibire i security tokens, i quali devono essere verificati al fine di provarne l’autenticità, l’integrità e il non ripudio, come descritto di seguito. Lo strumento fondamentale in grado di soddisfare tali requisiti è rappresentato dalla firma digitale. Inoltre, esistono diversi formalismi per descrivere i security tokens, come ad esempio le asserzioni SAML .

Integrità

I dati che vengono memorizzati nella piattaforma Cloud devono essere preservati da modifiche non autorizzate. Come già detto, la titolarità è uno degli aspetti da gestire per ostacolare tali manipolazioni, anche se, da sola, non è sufficiente. Infatti, è opportuno anche fornire meccanismi per garantire l’integrità di un documento, che permettano di verificare che il contenuto sia rimasto inalterato durante la trasmissione o memorizzazione. Anche in questo caso, è possibile usufruire di tecniche come il watermarking, che sono usate per dimostrare l’originalità di un documento non contraffatto (in particolare viene applicata una “filigrana digitale” al documento).
Un’altra tecnica di largo uso è, come già detto, quella della firma digitale. Dato un documento e una chiave di cifratura privata, viene generata una firma a partire dall’impronta del documento. Quando si vuole verificare

l'integrità del documento, la sua firma viene processata con una chiave di cifratura pubblica e se il risultato corrisponde all'impronta del documento allora quest'ultimo non ha subito alterazioni. Come la cifratura, anche il processo di firma digitale può essere gestito direttamente dall'applicazione, oppure affidata ad un servizio offerto dalla piattaforma Cloud, come in Signing Hub Errore. L'origine riferimento non è stata trovata..
La natura condivisa di soluzioni di Cloud hosting è spesso indicata come un rischio per la sicurezza informatica e per l'affidabilità della firma digitale. In ogni caso, l'industria del Cloud hosting è soggetta a numerosi standard di certificazione per la sicurezza che includono controlli regolari ed un impegno a mantenere aggiornati i sistemi.

Non ripudio

Come accennato in precedenza, un altro punto cruciale è fornire la prova incontestabile di avvenuta spedizione (non ripudio del mittente) e/o di avvenuta ricezione (non ripudio del destinatario) di un documento, al fine di evitare che un mittente e/o un destinatario possano disconoscere i dati trasmessi. Tecniche di cifrature a chiave asimmetrica e di firma digitale offrono una soluzione al requisito di non ripudio del mittente. Tecniche basate sulla spedizione di ricevute di ritorno offrono, invece, soluzioni al requisito di non ripudio del destinatario.

Confidenzialità

Le principali tecniche per garantire la confidenzialità dei dati sono quelle di cifratura, da impiegare sia quando i dati devono essere memorizzati sia quando vengono scambiati all'interno della piattaforma Cloud o tra la piattaforma e utenti remoti. Tali tecniche possono essere raggruppate in cifratura a chiave simmetrica e cifratura a chiave asimmetrica. Nel primo caso, esiste un'unica chiave per eseguire operazioni di cifratura e decifratura, mentre nel secondo caso esistono due chiavi distinte. Generalmente, gli algoritmi di crittografia simmetrica sono a livello computazionale più veloci e semplici di quelli a chiave asimmetrica, anche se hanno lo svantaggio di dover gestire in maniera sicura lo scambio della chiave. Pertanto, di norma, viene preferito l'uso di quelli simmetrici integrandoli con l'uso di quelli asimmetrici per lo scambio delle chiavi da usare per la cifratura dei dati scambiati.
Comunque, in sistemi distribuiti come il Cloud Computing, è stata individuata una problematica aggiuntiva che può ledere la confidenzialità dei dati, ovvero la data segregation. I dati nei sistemi Cloud sono, infatti, tipicamente memorizzati in un ambiente condiviso insieme a dati di altri utenti. Pertanto, una volta che i dati sensibili sono trasmessi al Cloud, è essenziale che essi siano mantenuti saldamente separati da quelli non sensibili.

Autorizzazione

Una volta che un soggetto ha fornito il proprio insieme di security tokens, si devono prendere le decisioni riguardanti l'autorizzazione per valutare se è possibile abilitare l'accesso alla risorsa richiesta. Questo

presuppone la definizione di apposite politiche di accesso per i vari utenti, o classi di utenti, che possono utilizzare le risorse, dati o funzioni, offerte dalle applicazioni sul Cloud.

Disponibilità

Le risorse offerte devono essere sempre disponibili per gli utilizzatori del sistema e le elaborazioni devono poter restituire sempre il risultato atteso. Possibili dinamiche intenzionali o non intenzionali, come attacchi informatici, fenomeni di saturazione delle capacità della piattaforma di Cloud Computing o guasti, non devono poter compromettere la disponibilità dei servizi e delle risorse.

Soluzioni a tale aspetto spaziano dall'introduzione di meccanismi di replicazione per tollerare i guasti, all'allocazione efficiente ed efficace di servizi e risorse nella piattaforma di Cloud Computing per evitare saturazione e all'utilizzo di Intrusion Detection Systems (IDS) per rilevare possibili intrusioni e applicare opportune contromisure. Mentre, supportare la replicazione in piattaforma Cloud non rappresenta un problema, le altre due soluzioni sono di più complessa realizzazione. Il problema della giusta allocazione di servizi e risorse nel Cloud assume una notevole complessità vista la scala e l'eterogeneità dei sistemi sottesi ad una piattaforma Cloud. In più, tale problema ha spesso una natura dinamica e deve essere di volta in volta risolto quando una nuova richiesta è sottoposta dagli utenti. Esso, infatti, spesso può essere formalizzato come un problema di ricerca operativa di tipo NP–completo. Pertanto, si rendono spesso necessarie delle opportune euristiche al fine di rendere risolvibile il problema. Attualmente, esistono varie soluzioni IDS, ma di fronte a nuovi scenari applicativi nel Cloud Computing, gli approcci esistenti presentano diversi problemi. In primis, l'operatore degli IDS dovrebbe essere l'utente, non l'amministratore dell'infrastruttura Cloud. In secondo luogo, in molti IDS esistenti devono essere introdotti meccanismi di estensibilità, gestione efficiente e compatibilità con la virtualizzazione basata sul contesto.

Infine, il Cloud Computing soffre di vari attacchi tradizionali, anche se spesso gli attacchi ad un'infrastruttura Cloud accadono dall'interno della stessa. In particolare, gli utenti autorizzati di un Cloud possono cercare di ottenere dei privilegi particolari e commettere frodi e divulgazione di informazioni a terzi (o modificare le informazioni intenzionalmente). Ad esempio, è stato documentato nel 2009 un attacco DoS interno contro Amazon Elastic Compute Cloud (EC2).

Audit

L'uso delle risorse, le possibili interazioni con il mondo esterno e le operazioni effettuate devono essere registrate in appositi sistemi di log, al fine di poter effettuare analisi nel caso si verifichino malfunzionamenti o violazioni dei requisiti di sicurezza e risalire alle cause. Questo aspetto assume una connotazione particolare nelle piattaforme di Cloud Computing, che sono costituite da molti nodi dislocati geograficamente.

In generale, non è ipotizzabile assumere una soluzione centralizzata dei log, che sarebbe poco efficiente in

termini di prestazioni ed affidabilità. Inoltre, è molto complesso effettuare l'audit della piattaforma tecnologica e dei processi interni dei fornitori. A tal proposito, si sta assistendo ad una certificazione dei processi secondo standard internazionali da parte di questi ultimi.
Un documento che fornisce un indirizzo su questo tema è stato prodotto dall'associazione americana degli auditor dei sistemi informativi ISACA [12].

Accordi sui livelli di servizio

I fornitori di servizi Cloud devono impegnarsi a garantire un livello minimo di servizio misurabile in base a parametri oggettivi, raggiungendo con il cliente un accordo in merito. In questa ottica, è possibile definire un apposito Service Level Agreement (SLA), ovvero un contratto attraverso cui si definiscono le metriche di servizio (ad esempio, qualità del servizio), che devono essere rispettate dal fornitore della piattaforma Cloud nei confronti degli sviluppatori/gestori delle applicazioni ospitate. Il SLA deve basarsi inoltre sulle funzionalità offerte dalle varie applicazioni che vengono eseguite su una piattaforma di Cloud Computing, come il monitoraggio degli utenti, e/o la gestione documentale dei loro dati. La qualità di servizio offerta dalla piattaforma Cloud deve essere continuamente monitorata ed è opportuno che sia verificato se essa rispetta puntualmente gli indicatori dello SLA sottoscritto. Questo monitoraggio deve avvenire sia per la monetizzazione del servizio offerto da parte del fornitore di servizi Cloud, sia per attestare il servizio di cui le applicazioni beneficiano. Limitatamente alla sicurezza, il SLA può contenere metriche di qualità come la disponibilità delle applicazioni, il livello di privacy e integrità e/o il numero di casi di violazione dei requisiti di sicurezza.
Con riferimento alla gestione della privacy, quest'ultima è una delle maggiori preoccupazioni per i potenziali clienti di servizi Cloud. In generale, sia i fornitori sia i potenziali utenti si confrontano con diverse regolamentazioni per la protezione dei dati, dove le inconsistenze tra le legislazioni nazionali rappresentano una significativa barriera alla vasta adozione del Cloud Computing. In aggiunta, il rispetto della privacy è diventato un metro di paragone fondamentale per la scelta tra diversi fornitori.
Nel 2012, la Cloud Security Alliance [5] ha deciso di definire delle linee guida per il rispetto della privacy e delle best practices specificando un formato standard per i cosiddetti Privacy Level Agreements (PLA), attraverso cui un fornitore dichiara il livello di privacy (protezione e sicurezza dei dati personali) che è fornito a un dato cliente. La natura dei PLA è simile a quella degli SLA, ma riferiti alla privacy dei dati. In generale, un PLA è un'appendice di una SLA.
Ulteriori criticità sulla gestione dei dati riguardano il luogo dove vengono fisicamente memorizzati e il loro trasferimento all'estero. Con riferimento a questi punti, le normative sulla privacy vietano il trasferimento dei dati nei paesi extracomunitari se esso non avviene secondo specifiche disposizioni. Pertanto, è opportuno specificare questi aspetti nei PLA stipulati tra il cliente, titolare del trattamento, ed il fornitore di servizi Cloud.

Analisi della piattaforma Social IoT

Tra le piattaforme analizzate è stata individuata una soluzione architetturale che prevede la realizzazione dell'interoperabilità tra i vari ecosistemi di cui una casa o una struttura è dotata, per gestire ed integrare apparecchiature differenti e di diversa complessità, nonché per rispondere alle diverse esigenze di connettività e di adesione al moderno paradigma *anywhere, anytime, anyhow*.
Dallo studio del paradigma e relativa soluzione open source disponibile e sviluppata nell'ambito della Social-IoT alliace *(www.social-iot.org)*, è stata identificata la specifica in termini di blocchi funzionali necessari per una piattaforma social-pervasive.

4.1 Criteri e policy per la condivisione dei dati

I requisiti chiave della piattaforma riguardano: decentralizzazione, distribuzione, scalabilità, real-time, intelligenza e dinamicità. La piattaforma attua tali requisiti adottando un modello di cooperazione basato sull'edge computing dove i servizi anziché essere posti nel Cloud sono allocati in prossimità di dove sono prodotti i dati. Questo approccio è particolarmente interessante nel contesto dell'IoT poiché consente di effettuare azioni in real-time sui dati in arrivo, tenendo conto dei limiti della banda disponibile. Usando questa strategia distribuita è possibile abbassare i costi e migliorare l'efficienza.

Tale piattaforma è in grado di supportare lo sviluppo di applicazioni utilizzando un modello di cooperazione dei dispositivi basato su quello degli agenti autonomi interagenti. Nella piattaforma, gli agenti sono in grado di mitigare i problemi dovuti alla mancanza di ragionamento e intelligenza dei sistemi IoT. Ogni cosa è connessa attraverso l'IoT ed è in grado di effettuare ragionamenti e assumere un comportamento intelligente. L'intelligenza nelle cose può essere raggiunta usando agenti software incapsulati nelle cose. Questo avviene

astraendo gli oggetti fisici con oggetti virtuali (OV) e utilizzando gli agenti come gestori degli OV con cui interagiscono attraverso le API fornite dagli OV. Gli agenti, essendo autonomi, sono in grado di avere un Behavior (comportamento) definito da un insieme di regole che è in grado di analizzare il flusso dei dati provenienti dagli oggetti e di conseguenza prendere decisioni e interagire attraverso scambio di messaggi con altri agenti che gestiscono altri oggetti virtuali. La comunicazione fra agenti e OV oltre che streaming può anche essere di tipo publish/subscribe attraverso pattern che sono istanziati nel contenitore a cui appartengono gli OV.

In particolare, per quanto riguarda l'integrazione a livello degli oggetti, il modello degli OV è ampliato da una componente sociale basata sul modello SIoT (Social IoT). A tal proposito il contenitore degli OV comprende una rappresentazione sociale degli oggetti attraverso l'aggiunta di un oggetto sociale (OS). Tale OS è in grado di funzionare come un mediatore in grado di creare da un parte, attraverso la piattaforma SIOT, una rete sociale degli oggetti che può essere navigata e essere utilizzata per la scoperta degli oggetti che attuano determinati gradi di socialità. Dall'altra parte, attraverso le API che vengono fornite è possibile per gli agenti effettuare delle richieste. L'identificatore dell'oggetto sociale, attraverso un meccanismo di traduzione dell'indirizzo è riportato nel dominio degli indirizzi dell'OV. In questo modo al livello degli agenti la componente sociale appare come un'estensione delle proprietà dell'OV che può essere interrogata con il medesimo indirizzo dell'OV.

Ulteriori funzionalità prevedono:

- un servizio di pagine gialle in grado di ricercare gli agenti in base alle loro funzionalità;
- la possibilità di creare/cancellare agenti;
- un meccanismo di acquaintances per definire vari modi con cui strutturare gli agenti
- meccanismi per la gestione della concorrenza in fase di attuazione attraverso di diverse tipologie di agenti utilizzanti il meccanismo dei thread;
- tecniche basate sulla swarm intelligence per raggiungere forme di consenso decentralizzate.

La piattaforma offre un protocollo RESTful che permette agli oggetti di inviare e ricevere dati e permette una interoperabilità con altre piattaforme; un server che funge da intermediario per le comunicazioni tra oggetti; una struttura dati per ciascun oggetto che lo descrive e tiene traccia dello stato dell'oggetto; una coppia di API Key (Read API Key e Write API Key) associata a ciascun oggetto per identificarlo e per permettergli di leggere o scrivere nella propria struttura dati; una engine interna che implementa le amicizie relazioni e costruisce il grafo delle stesse; una engine interna che effettua search & discovery delle risorse attraverso il grafo sociale.

Un elemento chiave riconducibile all'obiettivo di definire i "criteri e le policies per la condivisione e selezione dei dati da condividere" oggetto di questo task è il Social Object (Social-Obj) che può essere definito come la rappresentazione sociale di un Virtual Object (VO) della piattaforma pervasiva sopra descritta. Tale definizione

implica che vi è una corrispondenza biunivoca fra un Social-Obj ed un VO. Il Social-Obj va' visto come un processo indipendente dotato di un proprio life-cicle. Il Social-Obj mette in relazione un oggetto fisico ed un Virtual Object con la Piattaforma SioT. Le principali funzionalità sono:

- ricevere i risultati dei MAC Discovery effettuate dal dispositivo fisico, per mezzo del Social Sensor (un modulo software da implementare entro i dispositivi fisici per (i) permetter loro di rilevare la presenza nei dintorni di altri oggetti sociali e (ii) comunicare tale informazione al Social Object che poi la inoltrerà alla piattaforma, (iii) per localizzarsi, (iv) per inviare i dati prodotti e le informazioni sullo stato del dispositivo), e di comunicarli alla piattaforma,
- interagire con il Virtual Object, a cui è associato, per l'esecuzione di query su base sociale,
- effettuare le query sociali, richieste dal Virtual Object, interagendo con la piattaforma. Si ipotizzano dei sotto processi che effettuino operazioni di caching per rendere le query più performanti.

Altro elemento della piattaforma è il Social Object Container che è una struttura che include i vari oggetti sociali definiti, un cosiddetto Social Sensor Dispatcher per interfacciarsi al Social Sensor e un Social Hook Dispatcher che funge da interfaccia con il Virtual Gateway previsto dalla piattaforma pervasive.

4.2 Policy di accesso: privacy

Le policy di accesso e condivisione dei dati in applicazioni SIoT, siano esse di tipo classico che opportunistico, possono dar luogo ad importanti problematiche di sicurezza visto che l'informazione è sensibile ad attacchi alla privacy.

A titolo di mero esempio funzionale si pensi ad uno scenario di Opportunistic trading. L'utente Bob vuole comprare un volume "Harry Potter" tramite l'agente di trading opportunistico (OTA) in esecuzione sul proprio smartphone. Man mano che Bob si muove con il suo smartphone, la sua richiesta di trading viene condivisa con i nodi nelle sue vicinanze (formando, quindi, una community opportunistica). Poiché il range di movimento e i pattern di mobilità di Bob sono fissati (il numero di persone che incontrerà è quindi limitato), per incrementare il numero di nodi che riceveranno la richiesta di trading e velocizzare il processo di dissemination della richiesta, l'OTA utilizzerà altri nodi broker per l'inoltro della richiesta di Bob. Chiaramente, una tecnica social-based sarebbe la più adatta per selezionare i broker. Due giorni dopo, Alice, una book seller che si trova in un'altra zona della città, viene rintracciata dall'OTA e dai broker. Nello scenario appena descritto, informazioni sensibili quali la posizione dell'utente, i pattern di mobilità, le sue preferenze ricavate dagli online social network, ecc., potrebbero essere utilizzate all'interno dei protocolli di data dissemination.

In questo contesto di SIoT opportunistica (*Figura 10*) sarà quindi necessario proteggere la privacy dei partecipanti permettendo ai loro dispositivi una condivisione/inoltro dei dati affidabile. Le tecniche di data anonymization che vanno a nascondere l'identità degli utenti nella fase di inoltro dei dati, possono essere una delle modalità con cui risolvere questo problema, ma vi sono altre problematiche legate al concetto di opportunistico di cui tener conto.

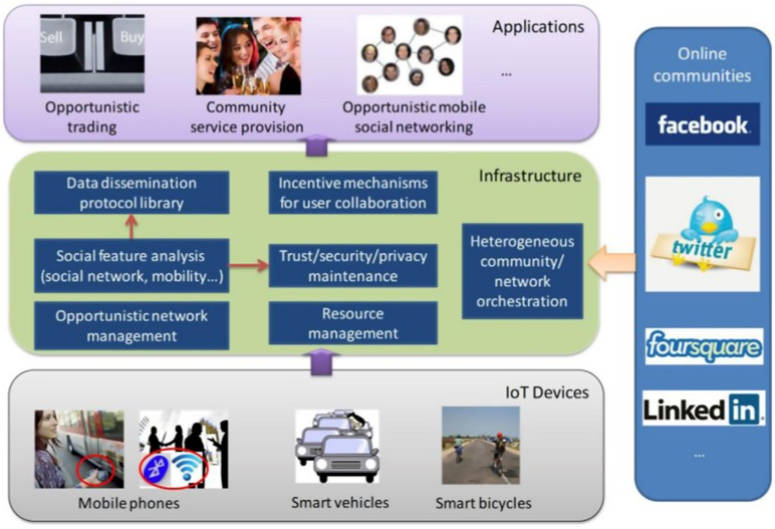

Figura 10 - Uno scenario di SioT opportunistica

I vari dispositivi IoT tramite un'infrastruttura di rete opportunistica che va a gestire risorse, aspetti sociali e di sicurezza, e i social network online, dissemina dati fra i vari utenti tramite varie applicazioni community-based.
La SIoT opportunistica offre un immenso potenziale ai consumer e ai service provider. Tuttavia, affinché queste innovazioni si traducano da idee in prodotti tangibili per il mercato di massa, vi sono altri aspetti da tenere in considerazione. Ad esempio, nei protocolli di data dissemination social-based, i broker dovrebbero offrire le loro risorse computazionali agli altri nodi della rete. La maggior parte dei dispositivi IoT (es. smartphone, sensori, ecc.), tuttavia, è caratterizzata da risorse limitate in termini di energia e di memorizzazione. Di conseguenza, si rende necessario lo sviluppo di strategie di incentivazione per la user collaboration per evitare che ciascun nodo riservi le proprie risorse solo per se stesso non partecipando al processo di routing.
La *Figura 11* mostra uno scenario opportunistico in cui è presente una situazione di questo tipo.

Inoltre, un nodo selfish potrebbe anche non instradare i dati altrui per paura di dati maliciuos da altre sorgenti o per mancanza di interesse nell'aiutare nodi appartenenti ad altre community, ad esempio. Questo tipo di comportamento porta inevitabilmente a messaggi ritardati o persi. Tali nodi necessitano dunque di essere individuati integrando nel modulo di data dissemination anche dei meccanismi di incentivazione che sfruttando la reputazione di un nodo premino i nodi che partecipano attivamente al forwarding.

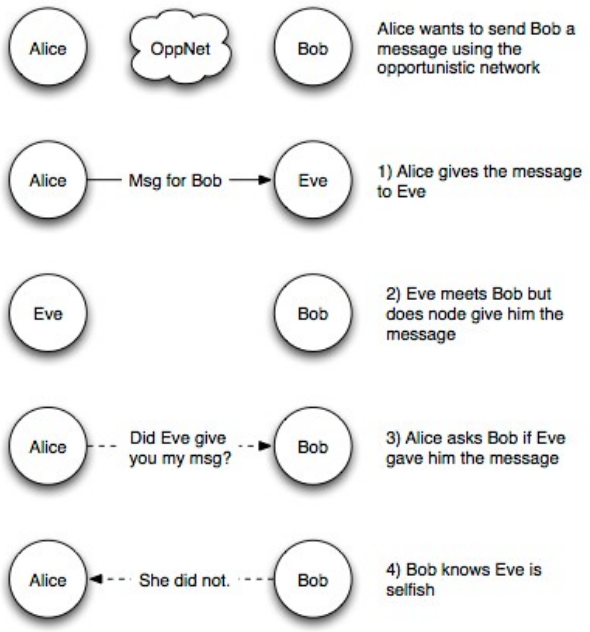

Figura 11 - Esempio di scenario opportunistico con selfish node detection.

In generale, la reputation rappresenta una stima di come un nodo o agente si comporterà in futuro basandosi su osservazioni del suo comportamento passato.

La reputazione può essere quindi sia l'accumulo di diverse osservazioni raccolte da vari agenti o il risultato dell'esperienza passata di un singolo agente. L'importanza di tale attributo è data dal fatto che grazie a questa informazione, i vari agenti all'interno dell'architettura SIoT proposta avranno la possibilità di prendere decisioni trust-based. Durante il corso di questa attività, analizzando approfonditamente lo stato dell'arte relativo a queste tematiche, si è visto che la maggior parte dei Reputation System (RS), implementati sia tramite architetture centralizzate che distribuite, utilizza i seguenti step:

1. **Information gathering.** Il primo step consiste nella raccolta delle informazioni dalle diverse entità del sistema. Questa informazione riguarda il comportamento passato visto dalle altre entità ed è un indicatore di quanto affidabile sia l'entità sotto analisi.
2. **Scoring and ranking.** Dopo aver raccolto e pesato le informazioni riguardo ad una entità, verrà calcolato un punteggio per questa entità basandosi su un algoritmo (es. fuzzy logic o Bayesian networks). Questo punteggio verrà poi inserito in una scala di ranking e rappresenterà un certo livello di trust che verrà utilizzato nella successiva fase di entity selection.
3. **Entity selection.** In questo step, un'entità ne seleziona un'altra (es. nella scelta di un next hop in un processo di un forwarding) basandosi sul suo punteggio. Solitamente, viene selezionata l'entità col punteggio migliore.

4. **Transaction.** A questo punto potrà esservi lo scambio di informazioni e servizi fra due entità (es. inoltro di un messaggio).
5. **Reward and Punish.** Durante l'ultimo step, le entità daranno un voto alla transazione basandosi sulla loro esperienza. In questo caso, l'esperienza è data da alcuni fattori sui quali alla transazione viene dato un voto.

L'Internet of Things, tuttavia, aggiunge delle nuove problematiche rispetto a quelle comunemente riscontrabili su RS online:

Eterogeneità. L'eterogeneità ha origine nel concetto di IoT. L'IoT interagisce con il mondo fisico con un grande numero di oggetti diversi che hanno in comune solo l'interfaccia con cui comunicare. I restanti componenti possono variare in base al dominio di un certo oggetto quali il sistema operativo, la connettività, i canali di I/O e le performance.

Scalabilità. Il numero sempre più crescente di oggetti connessi fra loro sta portando ad un crescente numero di comunicazioni, transazioni e dati. A causa di questa crescita, il RS dovrà essere scalabile rispetto al numero di dispositivi di rete per essere pienamente funzionale.

Infrastruttura. L'infrastruttura costituisce una problematica in termini di disponibilità e ricerca delle altre entità al fine di interagire con queste. Il RS deve necessariamente tenerne conto perché ciascuna entità ha bisogno delle altre per collezionare informazioni e interagire, e dovrà anche essere capace di trovarle all'interno della rete.

Identità. L'identità è una problematica comune alla classica Internet. La gestione dell'identità è una parte importante dell'IoT che deve essere presa in considerazione dal RS. Aspetti delicati di questa problematica riguardano l'identità di un oggetto che potrebbe non essere la stessa del meccanismo sottostante, gli oggetti che potrebbero avere una identità core e altre diverse identità, ed anche la possibilità che un oggetto decida di nascondere la propria identità.

Integrità. L'integrità è una problematica che non riguarda solo l'IoT ma ogni sistema che deve avere a che fare con hardware, software o dati. Il concetto di integrità assicura che una modifica non autorizzata al software, all'hardware o ai dati venga evitata, e che i dati siano internamente ed esternamente consistenti.

Risorse di rete. L'ultima problematica riguarda le diverse capacità di rete e le connessioni fra i vari oggetti. Questo significa che la disponibilità, la banda, e la latenza devono esser prese in considerazione specialmente per le interazioni time-dependent.

Sulla base delle problematiche individuate nella definizione di un RS per SIoT, è necessario definire l'architettura per il sistema di calcolo della *social reputation*. La social reputation è quindi definita stimando come un nodo/oggetto si comporterà in futuro basandosi su osservazioni del suo comportamento sociale passato derivato dalle sue interazioni sui social network e sui contatti wireless con gli altri oggetti.

I ratings e i punteggi (reputation o trust score) dei vari oggetti verranno gestiti in maniera distribuita come mostrato in *Figura 13* in tale scenario, gli oggetti dovranno essere capaci di rintracciare gli altri oggetti tramite

un algoritmo predefinito al fine ti trovare possibili transaction partner e reputation report. Gli oggetti memorizzeranno inoltre le esperienze dirette delle precedenti transazioni e se necessario, le esperienze di altri oggetti. La potenza computazionale degli oggetti dovrà anche esser tale da permettere di calcolare i valori di social reputation.

L'eterogeneità in tale architettura decentralizzata dovrà essere gestita con attenzione in quanto ogni dispositivo dovrà memorizzare e calcolare i reputation score da sé, incrementando così le necessità di risorse computazionali e di memorizzazione. Tuttavia, il calcolo del reputation score sarà tale da essere realizzabile nella maggior parte degli oggetti. In termini di scalabilità, l'architettura si presta ad un incremento dei dispositivi senza particolari problematiche vista l'assenza di un bottleneck. Per quanto riguarda invece le problematiche infrastrutturali e di risorse di rete, questo tipo di RS dovrà fare in modo che ciascun nodo possa rintracciare i nodi che non sono strettamente vicini. A tale scopo, potranno essere utilizzati degli approcci tipici delle reti peer-to-peer.

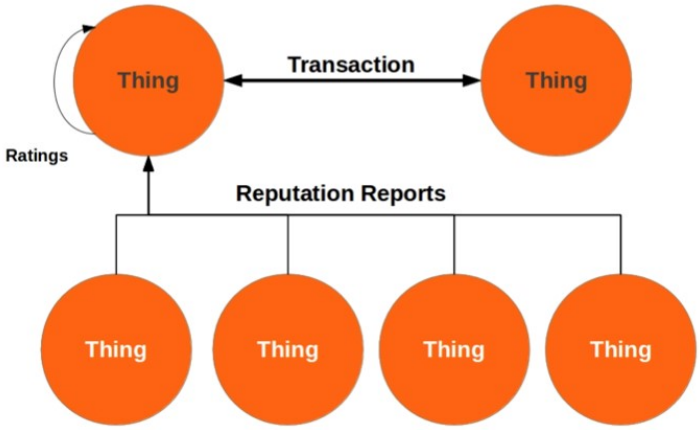

Figura 12 - Comunicazione decentralizzata nel sistema di social reputation.

Definizione dell'infrastruttura cloud che verrà utilizzata per l'implementazione della Piattaforma "Framework SHELL"

Nel presente paragrafo viene presentata l'infrastruttura CLOUD scelta per la piattaforma "Framework SHELL".

5.1 Descrizione della piattaforma Nuvola IT Self Data Center Telecom Italia

L'infrastruttura Cloud analizzata e scelta per il deploy del sistema cooperativo è Nuvola IT Self Data Center di Telecom Italia.

Il servizio Nuvola It Self Data Center, che è una soluzione di public CLOUD (SaS), nasce per fornire un accesso rapido e sicuro a risorse computazionali con la massima flessibilità di configurazione delle stesse. L'utilizzatore ha a disposizione una porzione di risorse della Nuvola Italiana, che può configurare in autonomia. Il servizio viene erogato su una piattaforma multitenant basata su suite di prodotti VMware. L'utilizzatore può in qualsiasi momento usufruire dell'accesso alla server farm, utilizzare i servizi di supporto e gli strumenti di self management.

Il servizio viene offerto in due modalità: *Pay-As-You-Go* e *Allocation Pool.*

La modalità *Pay-as-you-go* fornisce risorse secondo un modello full sharing (non ci sono risorse riservate) per lo specifico Cliente.

La modalità *Allocation Pool* fornisce risorse riservate al Cliente, con un ulteriore 33% di risorse elaborative (vCore e vRam) in full sharing per garantire la gestione di picchi di elaborazione del proprio Data Center Virtuale. Sono inoltre compresi il servizio di backup, l'antivirus e le licenze di sistema operativo Windows. Le caratteristiche principali della piattaforma Cloud sono:

- È una soluzione di Private Cloud Virtuale su infrastruttura fisica shared;
- È accessibile via web e permette di istanziare server virtuali, aggiungendo o diminuendo capacità elaborativa e storage;
- Si basa su una architettura completamente ridondata, sia in termini di infrastruttura fisica che di rete, oltre che basata su un hypervisor con servizi di alta affidabilità (HA) e Dynamic Resource Scheduling (DRS);
- È possibile riconfigurare le proprie Virtual Machine in maniera rapida e flessibile andando ad aggiungere o eliminare risorse (vcore, RAM e storage) da dedicare ai propri server;
- La piattaforma basata su VMware garantisce la compatibilità delle proprie applicazioni sia Windows che Linux evitando di dover modificare il software.

L'infrastruttura, vista come un unico pool logico indipendentemente dalla tipologia di architettura implementata, può essere gestita mediante un portale di servizio. L'utilizzo della suite software VMware combina l'agilità del cloud pubblico con la sicurezza, le prestazioni e la portabilità applicativa di cui le aziende hanno bisogno.

Il servizio Nuvola It Self Data Center è realizzato sulla base dell'infrastruttura cloud sicura di VMware attingendo da VMware vSphere, VMware vCloud Director, VMware vShield, VMware vCenter Chargeback e VMware vCenter Operation architettati e certificati da VMware.

Alcune caratteristiche dell'infrastruttura tecnica a supporto del servizio erogato all'interno della Nuvola Italiana di Telecom Italia:

- I Data Center si trovano sul territorio Italiano e offrono i più alti livelli di sicurezza fisica, sorveglianza presente 24 ore su 24, protezione perimetrale, locali e sale sistemi con alimentazione ridondata e sistemi di controllo all'avanguardia.
- L'utilizzo dell'hypervisor VMware garantisce completa ridondanza fisica e possibilità di failover, eliminando di fatto la possibilità di downtime per fault hardware.
- Il bilanciamento automatico dei carichi di lavoro (VMware DRS) garantisce le performance necessarie anche durante i picchi di utilizzo delle applicazioni.
- È previsto l'utilizzo di storage in SAN con meccaniche Fiber Channel e SATA.
- Per la connettività dall'esterno è possibile prevedere accessi internet e/o MPLS
- Utilizzo di server di classe enterprise.
- Tutte le risorse elaborative (server, SAN e storage) sono completamente ridondate abilitando così caratteristiche di prestazioni e affidabilità anche per le applicazioni più critiche.

L'architettura è riportata in *Figura 12*:

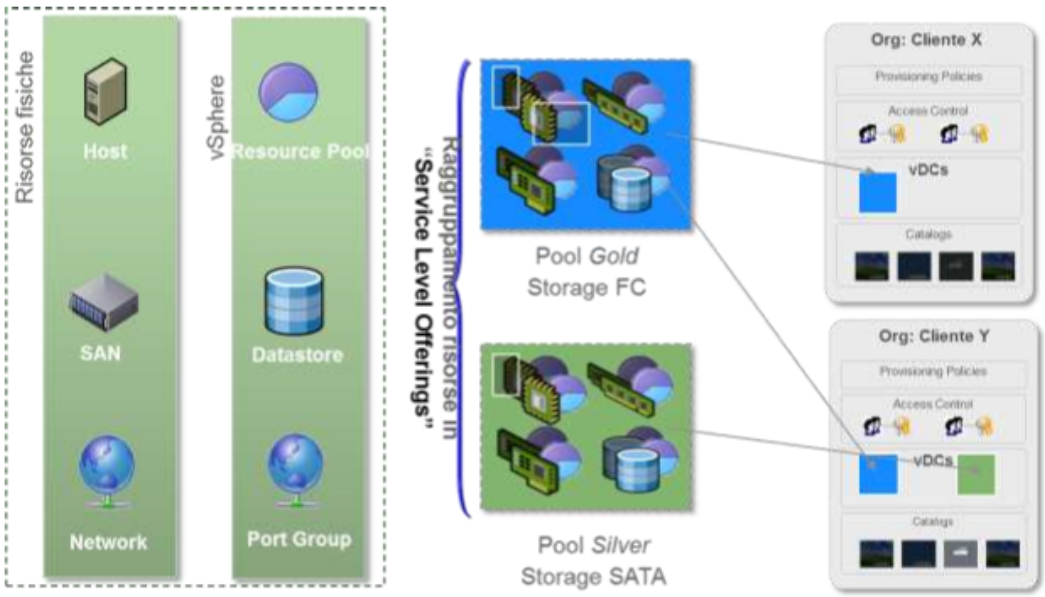

Figura 13 Architettura NUVOLA

Nella soluzione in esame, il modello di connettività prevede sia la rete Internet che MPLS che sono una "External Network" che corrisponde ad un Port Group su vSphere. L'accesso e l'erogazione del servizio avviene

di default mediante connettività Internet su banda condivisa

Gli IP pubblici previsti per l'accesso da Internet hanno già una serie di porte/servizi abilitabili sui firewall perimetrali del Data Center come indicato nella seguente *Tabella 2*:

Tabella 2 IP pubblici

source	port	type
internet	25/tcp 465 /TCP 691 /TCP	smtp
	53 / TCP 53 / UDP 123 / TCP 123 / UDP	DNS
	123/tcp 123/udp	NTP
	22/tcp 21/tcp 3389/tcp	ssh/sftp ftp rdp
	80 / TCP 8080 / TCP 443 / TCP 8443/tcp	http/https
	161/tcp 161/UDP 162/TCP 162/UDP	snmp
	389 /TCP 636 /TCP 379 /TCP 390 /TCP 3269 /TCP 3268 /TCP	LDAP LDAP/SSL
	143 /TCP 993 /TCP	IMAP4
	110 /TCP 995 /TCP	POP3

Nel progetto è previsto l'utilizzo di ulteriori porte/servizi riportati successivamente.

- ***MQTT***
  1883, 8883

- ***RabbitMQ***

  5672 e 5671 amqp

  25672 internode communication

  15672 web administration interface

  61613, 61614 (stomp client)

  15674, 15675 web sockets for stomp and mqtt

  Range 61000-62000 possibilmente aperto per testing di istanze multiple di rabbitMQ e framework Shell

- ***OpenVPN***
  TCP 943, UDP 1194 .

È disponibile il load balancer di piattaforma attraverso l'utilizzo di un virtual server che esegue il bilanciamento ad un pool di servers su uno specifico servizio. La configurazione di un pool inizia con la definizione dei servizi da bilanciare e delle service port utilizzate dai membri del pool. L'amministratore può selezionare tra servizi http, https e tcp. Ciascun servizio può utilizzare un diverso algoritmo di bilanciamento e gli algoritmi selezionabili sono: round-robin, URI e Least Connected. Possono inoltre essere configurati dei meccanismi di health check, si possono specificare pesi diversi per ciascun membro del pool ed inoltre ed è possibile specificare meccanismi di persistenza delle sessioni sulla base del protocollo utilizzato.

Per quanto riguarda i servizi di connettività interna è possibile connettere o isolare le VM all'interno del pool di risorse utilizzato.

Sono disponibili 2 tipi di rete:

1. **Diretta:** Accessibile da più organizzazioni. Le macchine virtuali appartenenti a organizzazioni differenti possono connettersi a questa rete e visualizzarne il traffico. Tale rete fornisce una connettività diretta a livello del layer 2 alle macchine esterne all'organizzazione, che possono connettersi direttamente a quelle interne. Questo tipo di rete viene fornita in combinazione con l'accesso MPLS
2. **Instradata:** Accessibile solo da una organizzazione. Solo le macchine virtuali incluse nell'organizzazione possono connettersi a tale rete. Questo tipo di rete fornisce anche accesso controllato a una rete esterna. Gli amministratori di sistema e gli amministratori dell'organizzazione possono configurare le impostazioni NAT (Network Address Translation), del firewall e VPN in modo da rendere determinate macchine virtuali accessibili dalla rete esterna.

In *Figura 11*, sono rappresentale le molteplici opzioni di connettività realizzabili all'interno dell'organizzazione e tra questa e il mondo esterno, che consentono la implementazione di landscape più o meno complessi.

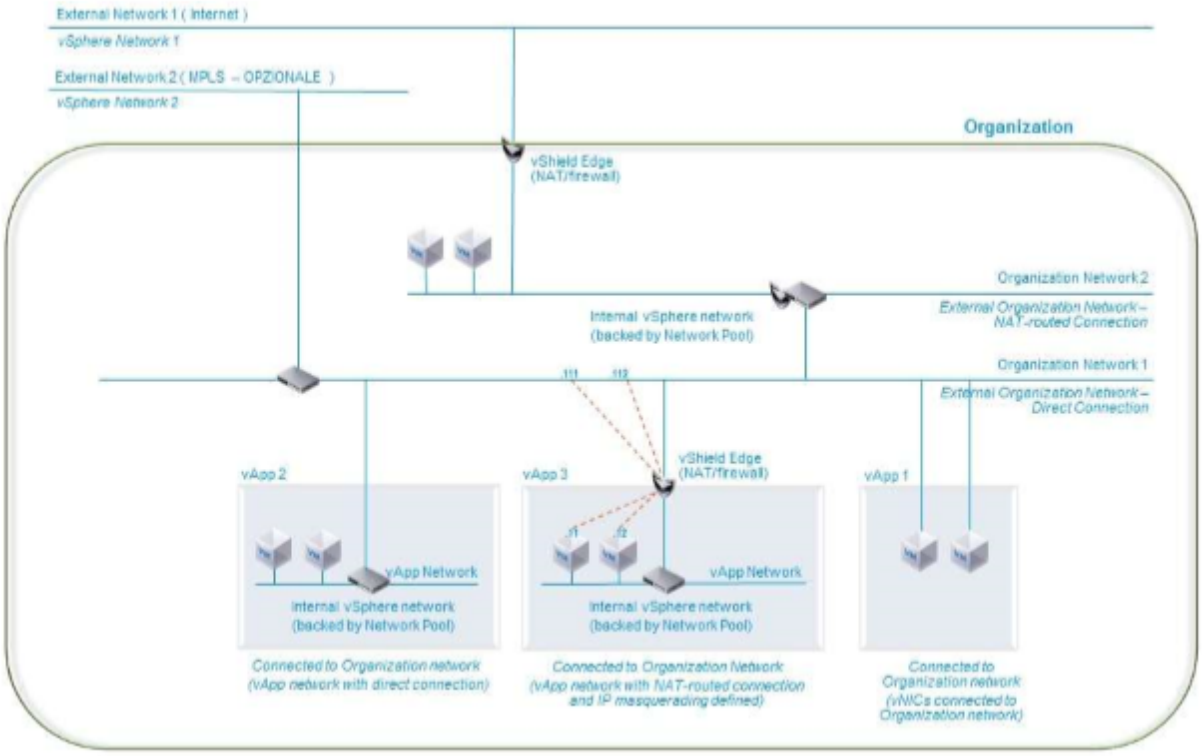

Figura 14 - Opzioni di connettività

All'interno dell'organizzazione, infatti, è possibile avere:

- vApp con VM isolate e VM connesse su una rete interna isolata (vApp 1)
- vApp con VM connesse a una rete interna, ruotata (vApp2) sulla Organization Network
- vApp con VM connesse a una rete interna ruotata e con meccanismi di natting e firewalling (vApp3)

L'Organization Network, a sua volta, è connessa alla rete esterna direttamente, ovvero con meccanismi di natting/firewalling implementati dalla componente vShield Edge.

La soluzione tecnologica alla base del servizio cloud prevede l'utilizzo della soluzione VMware vCloud Director ed altri componenti della suite Cloud Infrastructure Management di VMware con la quale è possibile implementare contesti di sicurezza e rete isolati associando a ciascuno di essi risorse CPU, RAM e Storage con la possibilità di diverse modalità di allocazione e livelli di servizio.

La suite software utilizzata per erogare il servizio è la seguente:

- VMware vCloud® Director™
- VMware vShield Manager™
- VMware vCenter Chargeback™
- VMware ESXi™ 5.0 (Update 2)
- VMware vCenter™ 5.0
- VMware vCenter Operation Manager™ 5.6

VMware vCloud Director si basa su vSphere aggiungendo costrutti logici che facilitano l'utilizzo in multitenancy delle risorse (*Figura 15*).

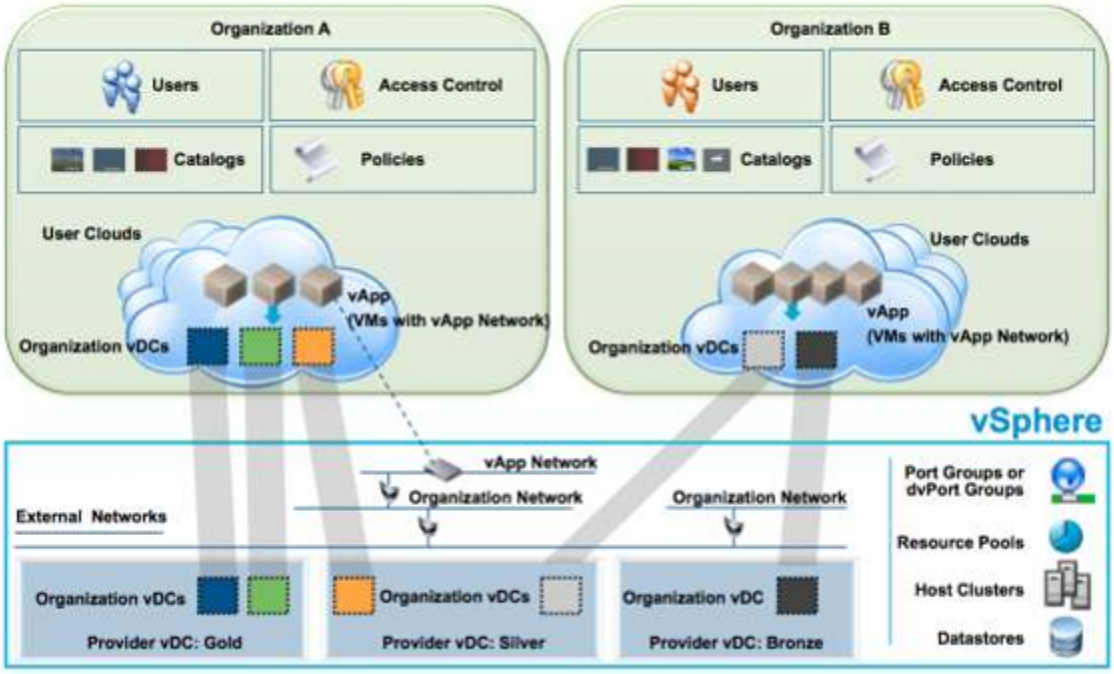

Figura 15 - VMware vCloud Director

In *Tabella 3*, un elenco dei costrutti di vCloud Director.

Tabella 3

Costrutto vCloud Director	**Descrizione**
Organization	Unità di amministrazione che rappresenta un raggruppamento logico di utenti, gruppi e risorse. Costituisce inoltre un confine di sicurezza entro il quale gli utenti possono attivare ed utilizzare Workload.
Provider virtual Datacenter	Insieme di risorse Server (Cluster vSphere o Resource Pool) e Storage fornite da un ambiente vSphere.
Organization virtual datacenter	Allocazione di un sottoinsieme di risorse fornite da un Provider virtual datacenter che sono assegnate ad un'Organization.
vApp	Contenitore di una o più macchine virtuali. In vCloud Director le VM sono sempre contenute in una vApp.
vApp template and media catalogs	Insieme di servizi disponibili per l'utilizzo. Un Catalog può contenere vApp Template e/o media

	(immagini ISO dei sistemi operativi).
Internal and external organization networks	Le reti virtuali che forniscono connettività di rete per vApp all'interno di un'Organization. Le Organization Network possono essere reti isolate utilizzate per la connettività tra vApp entro i confini dell'Organization (Organization internal network), o collegate a una rete esterna preconfigurata (VLAN – Port Group su vSphere), utilizzando una connessione diretta o NAT-Routed, per fornire la connettività al di fuori dell'Organization (External Organization Network). Alcuni tipi di Organization Network sono supportati da network pool.
vApp network	Rete virtuale contenuta all'interno di una vApp che consente la connettività di rete tra le macchine virtuali nella vApp. Una vApp Network può essere collegata ad una Organization Network utilizzando una connessione diretta o NAT-Routed per consentire la comunicazione con vApp nella organizzazione o all'esterno dell'organizzazione se l'Organization Network è connessa ad una rete esterna. Le vApp Network sono supportate da Network Pool. La maggior parte degli utenti con accesso ad un vApp può creare e gestire le proprie vApp Network.
Network pool	Un insieme di reti preallocate che vCloud Director può utilizzare per creare reti internal e NAT-Routed

Approfondiamo, ora, il concetto di catalogo e vApp.

Un catalogo è un contenitore per i template di vApp e media files (es. immagini ISO).

La piattaforma cloud fornisce un catalogo standard di vApp Template utilizzabili da tutti gli utilizzatori (Catalogo Servizi Pubblico) ed è inoltre possibile per i clienti creare in autonomia cataloghi all'interno delle proprie Organization così come eseguire upload di proprie immagini. iso ed installare VM completamente nuove con sistemi operativi propri. Nel catalogo pubblico sono inseriti i seguenti template di vApp (*Figura 16*):

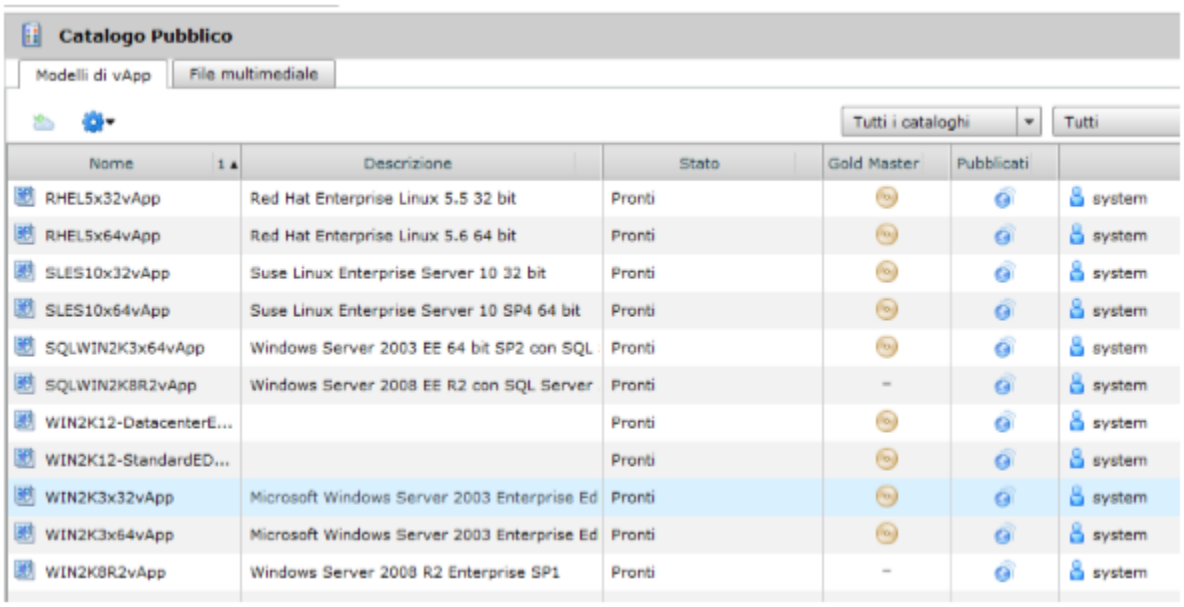

Figura 16 - Catalogo Pubblico

Partendo da questi template si possono creare le vApp/VM e successivamente modificare le risorse assegnate. Si possono, comunque, istanziare VM con altri Sistemi Operativi, purché compatibili con VMware, utilizzando proprie immagini ISO in locale.

Per quanto riguarda i profili degli utilizzatori è stato già detto che la soluzione si basa su una architettura multitenant. In *Figura 17* le caratteristiche dei ruoli predefiniti. Il System Administrator può creare nuovi ruoli a partire dai permessi disponibili.

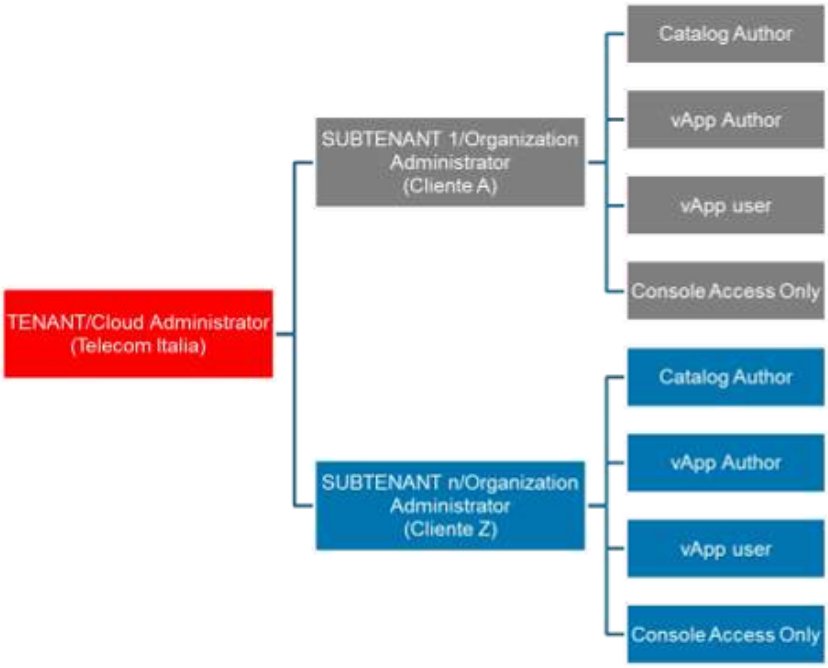

Figura 17 - Caratteristiche dei ruoli

TENANT/Cloud Admin (System Administrator). È il ruolo svolto dal Service Provider (Telecom Italia) ed è il "root user" dell'infrastruttura Cloud. È l'unico ruolo predefinito che ha permessi validi su tutti i costrutti vCloud Director e su tutte le Organization:

- Fa il deployment dell'infrastruttura e la gestisce
- Associa il vcenter server
- Crea i virtual Data Center (associa le risorse computazionali), le External Networks (fornisce l'accesso dall'esterno al pool di risorse) e i Network Pools (crea un insieme di reti di livello 2)
- Crea le Organizations
- Crea gli Organization VDCs (es. pool di risorse con storage FC) e le Organization Networks (per la connettività delle vApp all'interno di una organization)
- Può creare e gestire ruoli ed utenti

SUBTENANT/Organization Administrator. Con questa utenza l'utilizzatore può creare nuovi utenti e assegnare dei ruoli. È il service administrator lato utente ed è il "root user" dell'organization:

- Fa il deployment delle utenze e le gestisce
- Crea i cataloghi specifici per l'organization
- Gestisce le policy dell'organization (lease, quotas e limits) e la configurazione delle notifiche via mail

- Può visualizzare e configurare impostazioni Firewall/NAT/DHCP/VPN sulla Organization Network NAT-Routed

SUBTENANT/Catalog Author. Ruolo assegnato dall'Organization Administrator nella propria Organization:

- Crea cataloghi nella propria organization
- Può creare e gestire vApp modificandone le impostazioni

SUBTENANT/vApp Author. Ruolo assegnato dall'Organization Administrator nella propria Organization:

- Può creare e gestire vApp modificando anche parametri virtual hw delle VM
- Ha inoltre gli stessi diritti di un vApp User
- Può cancellare le proprie vApp
- Può eseguire operazioni su vApp: Start/Stop/Suspend/Reset ed accesso VM Console
- Può eseguire Copia/Move vApp

SUBTENANT/vApp User. Ruolo assegnato dall'Organization Administrator nella propria Organization, è il ruolo tipico del System Administrator per uno o più workload:

- Può cancellare (ma non creare) le proprie vApp
- Può modificare le impostazioni delle proprie vApp
- Può eseguire operazioni su vApp: Start/Stop/Suspend/Reset ed accesso VM Console
- Può eseguire Copy/Move vApp
- Modifica impostazioni VM tranne parametri Virtual Hardware

SUBTENANT/Console Access Only. Ruolo assegnato dall'Organization Administrator nella propria Organization, è il ruolo tipico dell'utente che deve interagire solo con il sistema operativo guest delle VM contenute in una vApp. Può visualizzare ed interagire con la VM console delle VM contenute in una vApp.

Infine, la piattaforma cloud mette a disposizione i seguenti strumenti:

- Web User Interface. È l'interfaccia grafica attraverso la quale si può gestire in autonomia il servizio.
- API (Application Programming Interface). È un insieme di procedure disponibili per eseguire task su vCloud Director mediante l'accesso ad un servizio REST esposto sulla stessa interfaccia web della UI di vCloud Director.

SICUREZZA

La sicurezza della piattaforma di erogazione dei servizi cloud di Telecom Italia è articolata su più livelli:

- **Sicurezza fisica degli ambienti di erogazione del servizio**. Data center costruiti con impianti all'avanguardia, sorveglianza presente 24 ore su 24, protezione perimetrale, locali interni e sale sistemi.
- **Certificazione ISO/IEC 20000:2005** per la Gestione dei Servizi Informatici. Rappresenta uno strumento di riferimento per l'organizzazione dei servizi informatici che mira al miglioramento dell'erogazione/fruizione dei servizi IT, ponendosi come obiettivo il raggiungimento della massima qualità dei servizi erogati e un massimo contenimento di costi.
- **Certificazione UNI EN 9001:2008** per attività di progettazione, sviluppo e attivazione di componenti infrastrutturali per l'erogazione di servizi cloud in modalità IAAS
- **Certificazione ISO14000**
- **Certificazione ISO27001** per i servizi di delivery, operation e sicurezza fisica dei DC di Telecom Italia
- Applicazione dell'**Information Security Management System** per la struttura IT Service Management in conformità ai requisiti standard ISO/IEC 27001 relativamente a
    - processi di Delivery ed Esercizio per l'erogazione di soluzioni Housing e Hosting nei DC di Telecom Italia
    - servizi di PdL Management, LAN Management e System Management nella conduzione dei sistemi per il mercato e per le pubbliche amministrazioni
- La Nuvola Italiana ha ottenuto la certificazione **CSA STAR**. La Cloud Security Alliance (CSA) è un'associazione internazionale che opera con lo scopo di promuovere l'utilizzo di best practice per la sicurezza del cloud computing, insieme alla formazione e sensibilizzazione nell'utilizzo sicuro del cloud
- **Sicurezza nell'accesso ai servizi dall'esterno**. I server di front end del portale di servizio sono collocati in una DMZ per fornire servizi all'esterno senza compromettere la sicurezza della rete aziendale interna oltre alla definizione di opportune policies sui firewall del Data Center
- **Sicurezza della suite sw VMware**. Relativamente alla piattaforma informatica ci sono diversi aspetti da tenere in considerazione
    - Le funzionalità di sicurezza del software VMware partono dalla piattaforma vSphere che segue i criteri per la certificazione EAL4+.
    - Anche per le funzionalità di rete, Firewall, VPN, Routing e NAT, la componente vShield segue i criteri per la certificazione EAL4+, fornendo funzionalità di firewall fra le varie vApp ed isolamento fra le Organization. I contesti di rete fra le diverse Organization sono isolati a livello L2.
    - L'accesso all'interfaccia web di vCloud Director ed alla Console delle VM avviene in maniera cifrata mediante SSL

Al riguardo la piattaforma cloud che sarà messa in esercizio per lo sviluppo della piattaforma di brokeraggio energetico analizzata sarà strutturata su tre pilastri fondamentali:

- un portale Internet come veicolo di accesso ai servizi di cloud computing, tramite internet browser o semplici client, opportunamente dotati di servizi di security;
- Banda larga di accesso a Internet;
- Infrastruttura innovativa di Next Generation Data Center che, grazie alla virtualizzazione di server, storage, applicazioni, desktop, permette all'infrastruttura IT della smart grid di essere flessibile e supportare richieste di variazioni, funzione delle necessità dei singoli prosumer e del soggetto aggregatore stesso, che consentirà di avere la modularità e la scalabilità necessaria al progetto.

Al fine di strutturare l'Architettura Cloud e procedere al dimensionamento dello stesso, decidere i sistemi operativi da utilizzare, gli ambienti di sviluppo e di eventuali middleware da mettere a disposizione sul Cloud, sono stati organizzati degli specifici incontri tecnici.

Sulla base di incontri tecnici e programmatici condotti con gli altri partner di progetto sono state individuate le effettive Virtual Machine da attivare al fine di rispondere a pieno alle logiche applicative e sistemistiche atte ad ospitare:

- Uno o più Data Base Server
- Un Media Server
- Diversi Application Server per la gestione dei vari servizi
- Le suddette scelte sono state derivate da analisi di natura tecnologica, di efficienza, modularità e flessibilità dell'architettura.

In termini di efficienza progettuale, pertanto, il gruppo di lavoro Telecom ha cercato di acquisire informazioni e competenze tecniche che secondo logiche di medio e lungo periodo, consentono di implementare azioni di:

i. Attivazione dinamica di risorse e servizi sulla base delle reali esigenze della Smart Grid
ii. Disimpegno dal continuo acquisto di risorse, con rapido dimensionamento in base in base alle necessità della rete
iii. Accesso alle risorse grazie a un provider esterno che consente di limitare i costi di investimento in capacità

infrastrutturale non utilizzata e in attività di consolidamento delle infrastrutture.

Di contro per quanto attiene al principio della Qualità si è voluto specificatamente garantire con gli studi di dimensionamento anticipati:

iv. delivery in grado di ridurre i tempi di risposta
v. up grading immediati di risorse HW e SW
vi. servizi "as a service", secondo strategie di attivazione in real time
vii. ridondanza dell'HW ed efficaci soluzioni di back up al fine di preservare e garantire in qualunque condizione la continuità operativa del sistema di Smart Energy System.

Riguardo alla Focalizzazione sui servizi di event processing ed alla correlata integrazione con metodi statistici e predittivi si è voluto inoltre offrire modelli di gestione end-to-end centralizzati ed una leva operativa delegata mediata da personale altamente specializzato e qualificato della stessa Telecom Italia, che gestisce le piattaforme Cloud,
Le peculiarità e la forte specializzazione e Know How di Telecom Italia sulla messa a disposizione e Gestione di Piattaforme Cloud, che erogano servizi cloud di nuova generazione, garantirà una alta affidabilità e qualità dei servizi erogati.

La Struttura caratterizzate la gestione ed Servizi Cloud vedono la presenza di:

i. Un Centro Nazionale di Assistenza (CNA),
ii. un Network Operation Center (NOC),
iii. una Control Room (CR) per amministrare centralmente l'infrastruttura,
iv. un Security Operation Center (SOC) a garanzia della sicurezza della rete, un Servizio di Assistenza e Consulenza.

Il profilo individuato per il dimensionamento della rete delle cose dovrà necessariamente attestarsi, secondo il gruppo di lavoro, su un Mid & Apps Management + KPI Premium dedicato nello specifico ad attori, come il soggetto aggregatore, che necessitano di elevati livelli di performance e di servizio per la gestione in hosting di ambienti mission critical, con particolare riguardo anche gestione del middleware o dei livelli dedicati alla virtualizzazione degli oggetti analizzati nelle presenti pagine.

In merito pertanto si è iniziato ad individuare i Server fisici x86 a uso esclusivo del Cloud delle Cose che consente di fatto di supportare specifiche esigenze con maggior riguardo ai concetti di scalabilità verticale.

E’ stata avviata anche l’attività di strutturazione della piattaforma Cloud e di un primo dimensionamento, per consentire di iniziare a sviluppare i servizi applicativi del Framework Shell.
Il dimensionamento legato al Cloud delle Cose ha inoltre spinto il gruppo di lavoro Telecom ad analizzare per ogni server virtuale la percentuale di potenza di CPU, garantita, a supporto di ognuno dei moduli applicativi in modalità shared al fine di garantire il massimo delle performance e delle elaborazioni,.
La massima velocità richiesta in termini applicativi e di elaborazione potrà essere di volta in volta ottimizzata in funzione del carico del nodo hardware, pur se virtualizzato, al fine di ottimizzare le performance hardware del server stesso.
In funzione del carico di lavoro anche le quantità necessarie di memoria fisica saranno assegnate sulla base di specifici rapporti di RAM per Core (1X, 2X, 4X) i cui valori sono determinati in base alle tecnologie messe a disposizione dei partner, ottimizzate in funzione delle esigenze predittive e di calcolo che sottendono al funzionamento della rete delle cose.
La soluzione ideata è inoltre caratterizzata da logiche di Load Balancing di Piattaforma con distribuzione del carico su più istanze server per conferire scalabilità e affidabilità dell'architettura del Cloud nel suo complesso.
Anche le strategie di backup a livello di sistema di Storage sono state ottimizzate al fine di garantire il ripristino in caso di fault, e a tali soluzioni sono state accostate e proposte ai partner del progetto strategie di Clustering votate alla creazione di un gruppo di istanze server, per conferire maggiore affidabilità e rendere l’architettura del cloud delle cose più performante.
I Service Element DBS (Data Base Server middleware), AS (Application Server middleware) e WS (Web Server middleware), sono stati proposti sotto forma di “template software” predefiniti con una configurazione iniziale standardizzata, basata sulle esigenze più comuni riscontrate negli ambienti già in esercizio.
Sul fronte DB Server sono stati proposti ed analizzati di fatto configurazioni ed architetture quali:

- DB Server MS-SQL (su piattaforma Windows)
- DB Server MySQL (su piattaforma x86 Linux-Windows)
- DB Server PostgreSQL (su piattaforma x86 Linux)

In termini di Application Server sono state analizzate le funzionalità e la stabilità di soluzioni Application Server sia x86 , che IIS su piattaforma Windows.

Nel corso del VI SAL, oltre ad avere scelto la Piattaforma Cloud per il Progetto SHELL, **il gruppo di ricerca di Telecom Italia** ha proceduto, dopo il confronto con tutti gli altri partner, al dimensionamento delle risorse da allocare Servizio Self Data Center come di seguito specificato nella *Tabella 2* :

Tabella 2 - Allocation Pool

Pool da 4 IP Pubblici	1
Assistenza	Standard
Portale di monitoraggio vRealize	Si
Numero Pool Base (6 GHz, 24 GB di RAM, 750 GB spazio disco FC)	0
Numero Pool Base SSD (6 GHz, 24 GB di RAM, 750 GB spazio disco SSD)	1
Blocchi di Upgrade di risorse elaborative (2 GHz, 8 GB di RAM)	1
Blocchi di Upgrade di CPU (2 GHz)	4
Blocchi di Upgrade di RAM (8 GB di RAM)	0
Blocchi di Upgrade di Storage FC (250 GB)	0
Blocchi di Upgrade di Storage SSD (250 GB)	1
Blocchi Storage NAS (500 GB)	0
Pacchetti licenze MS Remote Desktop Services (10 utenti)	0

All'interno del profilo **Allocation Pool** sono compresi:

Risorse oltre soglia per la gestione dei picchi;

Licenze S.O. Windows;

Antivirus;

Backup Base.

Nella *Tabella 3* è riportato il valore complessivo di risorse previste.

Tabella 3 - Risorse previste

GHz riservati	RAM riservata (GB)	Storage FC riservato (GB)	Storage SSD riservato (GB)
16	32	0	1000
GHz non riservati per gestione picchi	**RAM non riservata (GB) per gestione picchi**	**# IP Pubblici ***	
7	11	4	

* di cui 1 utilizzato per servizi di "routing" e quindi non direttamente utilizzabile dal Cliente

Nel periodo di riferimento è stato avviato il delivery del servizio Nuvola IT Self Data Center al fine di rendere

disponibile le risorse di cui sopra, per il secondo semestre. L'attività successiva per i prossimi mesi, prevenderà il supporto al partner UNIVPM per la predisposizione configurazione delle varie macchine virtuali e server virtuali da dedicare alla Piattaforma "Framework SHELL" ed ai Servizi da implementare.

Riferimenti

[1] Amazon Web Services http://aws.amazon.com/

[2] Amazon Elastic Compute Cloud http://aws.amazon.com/ec2/

[3] Amazon Simple Storage Service http://aws.amazon.com/s3/

[4] https://cloudsecurityalliance.org/

[5] Cloud Security Alliance – Italy Chapter: Cloud Computing – Benefici, rischi e raccomandazioni per la sicurezza delle informazioni, Marzo 2013

[6] ERCIM NEWS: Special Theme: Cloud Computing. October 2010

[7] The Eucaplyptus Cloud http://www.eucalyptus.com/eucalyptus-cloud/iaas

[8] http://www.gartner.com/it-glossary/about.jsp

[9] http://www.gartner.com/it-glossary/cloud-computing/

[10] Google Apps for Work http://www.google.com/apps

[11] C. N. Hoefer, G. Karagiannis: Taxonomy of Cloud Computing services. In Proceedings of GLOBECOM Workshops (GC Wkshps), 2010 IEEE, pages 1345 - 1350

[12] http://www.isaca.org/knowledge-center/research/researchdeliverables/pages/cloud-computing-management-audit-assurance-program.aspx

[13] Editors: Keith Jeffery, Burkhard Neidecker-Lutz. European Commission, Expert Group Report: The future of Cloud Computing. Opportunities For European Cloud Computing Beyond 2010

[14] P. Mell, T. Grance: The NIST Definition of Cloud Computing. Recommendations of the National Institute of Standards and Technology (NIST), U.S. Department of Commerce, September 2011

[15] OpenNebula http://www.opennebula.org

[16] Sun Microsystems: Introduction to Cloud Computing Architecture. White Paper, 1st Edition, June 2009

[17] L. M. Vaquero, L. Rodero-Merino, J. Caceres, M. Lindner: A Break in the Clouds: Towards a Cloud Definition. ACM SIGCOMM Computer Communication Review, Volume 39, Number1, January 2009

[18] Windows Azure http://azure.microsoft.com/