CLOUD COMPUTING EVOLUZIONARIO E RIVOLUZIONARIO

Mauro Migliardi
Roberto Podestà

Dietro al termine "Cloud Computing" si cela un elevatissimo livello di ambiguità e non esiste, ad oggi, una definizione univoca e universalmente accettata di ciò che è e ciò che non è Cloud Computing. In questo articolo cercheremo di chiarire alcuni degli aspetti più critici di questo concetto pur senza pretendere di dissipare completamente, e una volta per tutte, la nebulosità che ancora ammanta l'aggregazione delle tecnologie che collaborano a formare il Cloud Computing e impedisce di ottenere un'unica definizione.

3.3

1. INTRODUZIONE

Nel mondo variegato dei termini di successo, Cloud Computing è sicuramente uno dei più forti contendenti per la corona di campione del momento e, come si può osservare nella figura 1, esso ha recentemente soppiantato il Grid Computing tra i termini più "cercati" sul web. Allo stesso tempo, un tale sorpasso è ancora ben lontano a livello di pubblicazioni scientifiche, infatti, mentre una ricerca degli articoli su Grid Computing riporta oltre 4000 citazioni in CiteSeer^{x}, la ricerca per Cloud Computing si ferma a soli 83 articoli. Questo fornisce una prima ragione della quantità di ricerche *on-line* relative al termine Cloud Computing, ma è altresì vero che questa fame di informazione, deriva anche dal fatto che Cloud Computing viene oggi disinvoltamente presentato come la summa di una pletora di parole chiave nel panorama recente dell'*information technology*. Tra le più spesso utilizzate, basti ricordare *Service Oriented Architecture* (SOA), *application service provision*, Web 2.0, *Web Services, mash-ups, Utility Computing, Autonomic Computing, On-demand Computing* ecc..

Seppure in mezzo a questa doccia mediatica di parole chiave esista uno spazio per genuine opportunità di miglioramento dell'odierno uso commercial-industriale dell'*information technology* [1], è altresì indubbio che dietro al termine Cloud Computing si celi ancora un elevatissimo livello di ambiguità e una serrata lotta per rendere il termine sinonimo di una tra le diverse accezioni in uso ad oggi. Di fatto, non esiste ad oggi una definizione univoca ed universalmente accettata di ciò che è e ciò che non è Cloud Computing e seppure diversi gruppi e istituzioni si siano dedicati a questo difficile compito [2, 3], un reale consenso che vada oltre al vago riferimento all'astrazione delle risorse tramite interfacce standardizzate fornito da Wikipedia, ancora manca.

Il termine Cloud Computing deriva dall'uso comune nel campo dell'Ingegneria dell'Informazione di rappresentare Internet come una nuvola che tutto interconnette celando completamente l'infrastruttura estremamente complessa che tale servizio richiede. Allo stesso modo, il termine Cloud Com-

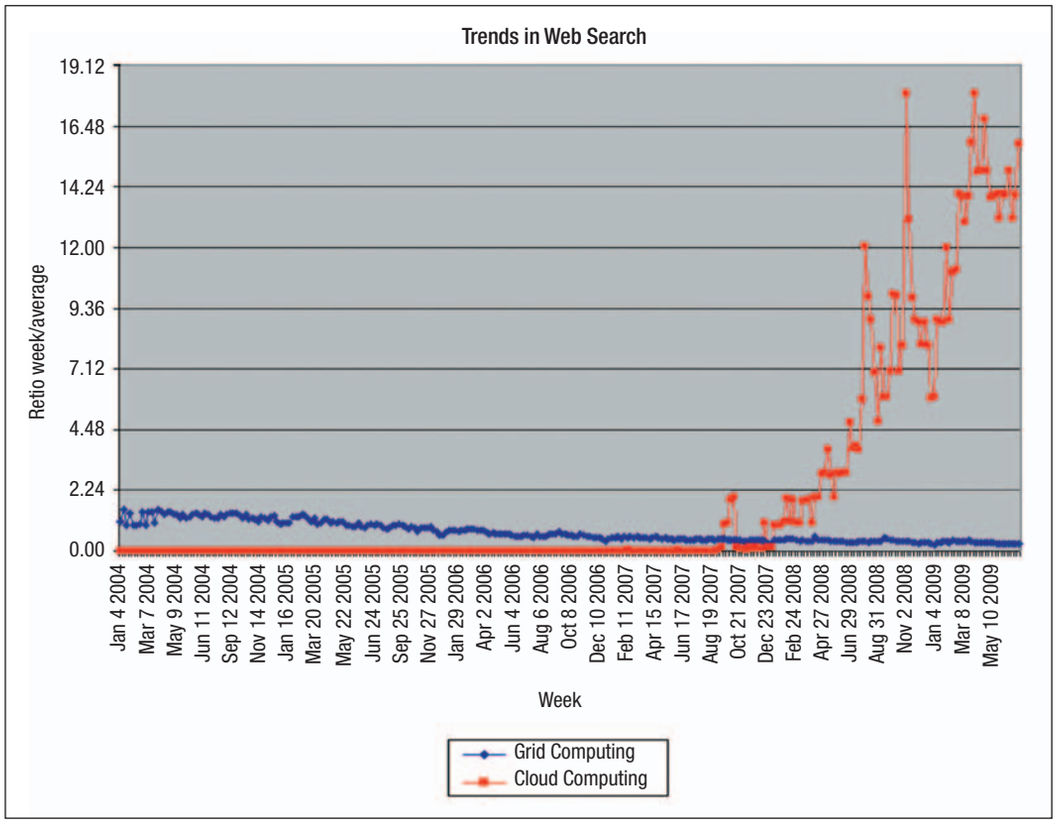

FIGURA 1
Trend esibito dalle ricerche effettuate tramite Google per i termini Grid Computing e Cloud Computing

puting implica lo spostamento del meccanismo principale di fornitura dei servizi computazionali dalla periferia, completamente controllata e posseduta dall'utente, verso una "terra incognita", dominio di differenti fornitori di servizi, nascosta dalle nebbie (o nuvole, appunto) e percepibile solo per mezzo di una semplice interfaccia di uso il più possibile standardizzata.
Come è facile vedere, un tale livello di astrazione si presta a generare molte e spesso tra loro non compatibili definizioni di Cloud Computing, tanto è vero che esiste ancora discussione se il termine Cloud si debba applicare ad una ed una sola infrastruttura (come una e una sola Internet) oppure se possano esistere molteplici Cloud eventualmente capaci di interazione.
In questo articolo cercheremo di chiarire alcuni degli aspetti più critici del concetto di Cloud Computing pur senza pretendere di essere noi a dissipare completamente e una volta per tutte la nebulosità che ancora ammanta l'aggregazione delle tecnologie che collaborano a formare il Cloud Computing e impedisce di ottenere un'unica definizione.

2. TASSONOMIA E DEFINIZIONI

Il Cloud Computing eredita sicuramente alcuni dei più importanti concetti nati in seno al Grid Computing [4, 5] nell'ultimo decennio. Tali concetti sono sviluppati e integrati a tal punto da poter considerare - almeno sotto certi aspetti - il Cloud Computing come un fenomeno evolutivo del Grid.
Tuttavia, sotto l'ombrello, a volte fin troppo ampio, del termine Cloud Computing vi sono anche aspetti genuinamente rivoluzionari per il panorama delle infrastrutture di *information technology* oggi alla base della gestione quotidiana delle aziende.
Come afferma Wolfgang Gentzsch in [6], le Grid in molti casi non sono state in grado di mantenere le ottimistiche promesse presenti nel libro blu [7] pubblicato ormai due lustri orsono. Tuttavia, affermare, come si fa in [8] che il Grid Computing (e più in generale il calcolo distribuito prima delle Cloud) richiede scrivere programmi basati su scambio di messaggi (riquadro 1 a p. 18) mentre il Cloud Computing supera tale complessità basandosi, ad esempio, su Map Reduce [9] è - per adottare un gentile eufemismo - catastroficamente fuorviante. Infat-

RIQUADRO 1 - Modello a scambio di messaggi

Il modello computazionale a scambio di messaggi viene introdotto negli anni '80 da C. A. R. Hoare. Esso si basa fondamentalmente su due primitive, *send* e *receive*, che permettono a processi strettamente sequenziali di comunicare e sincronizzarsi fra loro al fine di costruire un'applicazione concorrente o parallela di qualsivoglia complessità. Il modello a scambio di messaggi è quello più comunemente usato nel calcolo distribuito in quanto non richiede strutture complesse come la memoria condivisa, ma pone invece come unico vincolo architetturale la presenza di un canale di comunicazione tra tutte le componenti del sistema di calcolo.

ti, un'affermazione di questo tipo confonde un singolo strumento di scrittura di codice parallelo (lo scheletro algoritmico *Map-Reduce*) con la sottostante struttura di gestione delle risorse di calcolo e maschera la semplificazione proveniente dall'affrontare una specifica tipologia di problema come una mutazione epocale. È del tutto evidente che il problema della scrittura di codice massicciamente parallelo è uno degli scogli con cui si è scontrato il Grid Computing, tuttavia, affermare che ciò che distingue Grid Computing da Cloud Computing è il limitarsi ad affrontare problemi che possono essere risolti con una tecnica di programmazione che, tra l'altro, come dice lo stesso articolo in cui venne presentata [9], è applicabile in diversi casi reali ma non in generale in quanto richiede, per essere efficiente, che i *task* mappati siano *embarassingly parallel*, è, come minimo, semplicistico e non fornisce alcun reale strumento per dipanare la matassa di cosa sia il Cloud Computing.

Seppure un vero consenso sulla definizione di Cloud Computing non sia stato ancora raggiunto, e il *white paper* su Cloud Computing [10] recentemente prodotto dal gruppo *Reliable Adaptive Distributed Systems* presso l'University of California Berkeley neghi questa tassonomia, vi è una generica convergenza sulla possibilità di definire **tre principali livelli di visione della nuvola**.

Il primo livello è quello delle applicazioni in-Cloud (*Application as a Service*), il secondo livello è quello delle piattaforme in-Cloud (*Platform as a Service*) e il terzo livello, quello più vicino all'hardware, è quello dell'infrastruttura in-Cloud (*Infrastructure as a Service*).

Questi tre livelli possono essere considerati tre punti su di un asse che ha come coordinata il livello di gestione dell'applicazione fornito dalla Cloud (Figura 2).

❑ **Il primo livello**, quello delle applicazioni in-Cloud è, di fatto, quello che viene da anni chiamato *Software as a Service* (SaaS). Seppure esistano esempi innovativi di questa tipologia di Cloud (e.g. il software CRM di SalesForce.com), questa tipologia di cloud è ormai parte della panoplia di tutti i professionisti IT. Servizi come gmail e google maps sono applicazioni in-Cloud che tutti conosciamo e che la maggior parte di noi utilizzano. La scalabilità, per la maggior parte di queste applicazioni risiede esclusivamente nella struttura di accesso e nella capacità di manipolare grandi moli di dati non necessariamente in relazione tra loro. Si pensi, per esempio, a gmail: un'architettura basata su di una macchina virtuale per utente è, in linea di principio, in grado di soddisfare le esigenze del servizio ed è solo la dinamicità con cui muta il numero di utenti che richiede un'effettiva elasticità dell'infrastruttura. In questo caso, il gestore della Cloud possiede sia l'infrastruttura che l'applicazio-

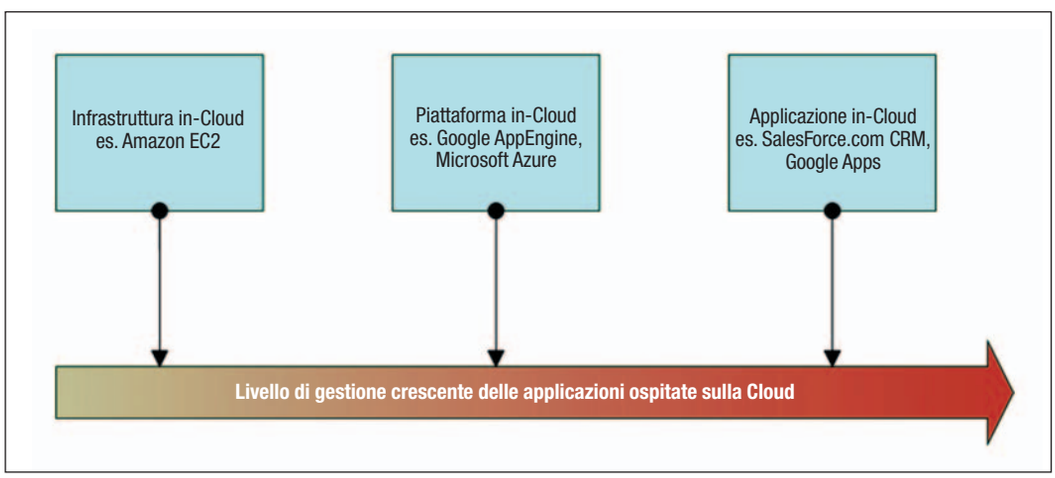

FIGURA 2
Asse del livello di gestione dell'applicazione fornito dalla Cloud

ne che viene venduta sotto forma di servizio accessibile via Web.

❑ **Il secondo livello**, quello delle piattaforme in-Cloud, è lo sviluppo più recente del Cloud Computing. In questo livello possiamo inserire Google AppEngine [11], Microsoft Azure [12] e Amazon Elastic MapReduce [13]. Questa tipologia di Cloud (Figura 2) si trova a metà strada tra una gestione delle applicazioni totalmente a carico della cloud come nel primo livello ed una totalmente a carico dell'utente come nel terzo livello. La caratteristica principale di questo tipo di nuvola è quella di fornire delle interfacce per la programmazione delle applicazioni (API[1]) specifiche secondo le quali sviluppare applicazioni orientate ad essere usufruite come servizi via web ed un'interfaccia web per ottenere l'installazione di tali applicazioni sulle risorse computazionali della Cloud in modo che il livello di risorse necessarie per mantenere una qualità di servizio definita possa essere deciso, fornito e gestito dalla Cloud stessa in base al *Service Level Agreement* (SLA) stipulato dallo sviluppatore. In questo caso, il gestore della Cloud possiede l'infrastruttura ma non il codice dell'applicazione resa disponibile via Web. L'applicazione e i proventi ottenuti tariffando il servizio fornito dall'applicazione sono appannaggio dello sviluppatore dell'applicazione. L'unico vincolo che il gestore della Cloud pone sull'applicazione è la tecnologia da utilizzare per sviluppare questa applicazione. Si può citare, ad esempio, la dipendenza da .Net della piattaforma in-Cloud Azure di Microsoft e la dipendenza dalle API di Google Datastore per Google AppEngine.

❑ Infine, **il terzo livello**, quello dell'infrastruttura in-Cloud, trova il suo esponente più noto in *Elastic Computing Cloud* (EC2) di Amazon [20]. In questo caso, il gestore della Cloud possiede solo l'infrastruttura di calcolo e non pone alcun vincolo né sul tipo di applicazione che l'utente eseguirà, né sulla tecnologia da lui utilizzata per sviluppare tale applicazione. In questa situazione, la virtualizzazione (riquadro 2) ha due ruoli chiave: il primo è, ovviamente, quello di permettere la rapida fornitura e messa in linea di nodi computazionali su richiesta degli utenti, ma il secondo, in questo caso forse ancora più importante, è quello di difendere l'infrastruttura e le applicazioni degli altri utenti dal comportamento volutamente o accidentalmente dannoso dell'applicazione. Questa difesa è fornita dal totale isolamento in cui una macchina virtuale incapsula la o le applicazioni in esecuzione su di essa. Infine, un vantaggio in un certo senso collaterale, ma certo non marginale dell'isolamento fornito dall'utilizzo della virtualizzazione è la possibilità di configurare l'ambiente di esecuzione di un'applicazione secondo i desideri dell'utente senza dover in alcun modo temere che specifiche versioni di sistema operativo o di librerie possano collidere con le necessità di altri utenti.

RIQUADRO 2 - Virtualizzazione e Paravirtualizzazione

Per virtualizzazione si intende la creazione di una versione virtuale di una risorsa normalmente fornita fisicamente. Qualunque risorsa hardware o software può essere virtualizzata: sistemi operativi, server, memoria, spazio disco, sottosistemi. Un esempio base di virtualizzazione è la divisione di un disco fisso in partizioni logiche.

Meccanismi più avanzati di virtualizzazione permettono la ridefinizione dinamica tanto delle caratteristiche della risorsa virtuale, quanto della sua mappatura su risorse reali.

Per poter parlare di virtualizzazione è necessario introdurre il concetto di macchina virtuale o *virtual machine*. In origine, il termine "*virtual machine*" veniva usato per indicare la creazione di una molteplicità di ambienti di esecuzione identici in un unico computer, ciascuno con il proprio sistema operativo, infatti, questo genere di virtualizzazione è particolarmente utilizzata nel campo dei mainframe e dei supercomputer. Lo scopo di tale tecnica era, originariamente, quello di sezionare tra più utenti l'uso di un singolo computer dando ad ognuno l'impressione di esserne l'unico utilizzatore.

L'uso più comune oggi è quello di generare ambienti "sterilizzati" e completamente separate tra loro per poter, per esempio, ricostruire l'ambiente necessario all'esecuzione di un programma (in termini di sistema operativo, specifiche librerie ecc.) senza porre vincoli sulle alter applicazioni se non quello di poter usufruire di una quota parte delle risorse globali. Questo risultato è oggi ottenuto per mezzo di un programma che emula un calcolatore; è possibile, infatti, realizzare tramite software, un finto computer su cui può essere eseguito un sistema operativo diverso da quello che equipaggia realmente l'elaboratore.

La virtualizzazione può essere realizzata secondo modalità diverse, tra cui, principalmente, citiamo:

- emulazione: la macchina virtuale simula completamente l'hardware, utilizzando un sistema operativo standard che viene eseguito dalla CPU virtuale;
- paravirtualizzazione: la macchina virtuale non simula un hardware completo, ma offre speciali API che richiedono modifiche nel sistema operativo ma forniscono prestazioni superiori.

3. UN INQUADRAMENTO "STORICO" DEL FENOMENO CLOUD COMPUTING: EVOLUZIONE...

Come accennato, molto spesso, il Cloud Computing è visto come un'evoluzione del Grid Computing. È però importante notare una di-

[1] *Application Programming Interface*.

stinzione a livello concettuale tra i due "fenomeni". Secondo gli obiettivi iniziali definiti più di dieci anni fa dai padri del Grid, Ian Foster e Carl Kesselmann, si sarebbe andati verso un'infrastruttura computazionale in grado di far leva sulla potenza di calcolo inutilizzata delle macchine in stato idle genericamente connesse a Internet. In pratica, nell'accezione originaria del Grid con la famosa analogia tra la rete elettrica e la potenza di calcolo che diventa slegata dalla localizzazione fisica dell'utente, si tenderebbe ad avere dei "micro produttori" di "computing" aggregati in un'organizzazione virtuale attraverso un *middleware* capace di fornire la visione di un unico supercomputer virtuale in cui l'utente stesso condivide le proprie risorse di calcolo. Al contrario, il Cloud Computing riporta a un'idea di "macro produttori" di "*computing*" indipendentemente dal tipo di cloud (*application, platform* o infrastruttura). Infatti, l'utente utilizza la Cloud fornita dal provider che, generalmente, si tratta di un centro di calcolo, che può anche essere frutto dell'aggregazione di centri sparsi per il globo, di proprietà esclusiva del provider.

Questa differenza concettuale è però molto sfumata a causa dell'evoluzione del Grid Computing che si è avuta a partire dalla fine degli anni '90. Infatti, i principali risultati e progetti più significativi, in alcuni casi diretti dagli stessi Foster e Kesselmann, che hanno finito per essere considerati propriamente il Grid Computing, hanno riguardato l'aggregazione di centri di supercalcolo che, guarda caso, solitamente sono l'ossatura delle Cloud. Prima di evidenziare questa somiglianza vista come linea evolutiva, per completezza di informazione, va detto che il concetto originario di Grid Computing non è andato completamente smarrito. Sviluppi in questo senso seppur considerati marginali rispetto alla corrente principale sono le Desktop Grid che hanno avuto come precursori seti@home (e.g. BOINC, XtreamWeb e il progetto europeo EDGeS) e quelli derivanti dalla convergenza tra Peer To Peer e Grid Computing [23] come WOW [25].

Riprendendo la linea evolutiva, l'utilizzo di tecnologie Web-based per l'accesso a un'infrastruttura di calcolo distribuita e l'utilizzo della virtualizzazione dell'hardware come base per l'infrastruttura di calcolo distribuita sono i principali aspetti alternatamente presenti nelle principali realizzazioni di Grid Computing antecedenti alla comparsa del Cloud Computing che inducono a vedere questo fenomeno come un'evoluzione che raccoglie e armonizza in un'unica infrastruttura/piattaforma questi aspetti tecnologici.

La convergenza tra Grid Computing e *Web Services* è stata ampliamente discussa e investigata già a partire dagli albori di queste due tecnologie. Tale sinergia si è unanimemente articolata nell'attribuire ai *Web Services* e, più astrattamente al modello *Service Oriented Architecture* (SOA), un ruolo chiave nell'accesso alle griglie computazionali.

La visione orientata ai servizi e l'utilizzo di tecnologie aperte e altamente standardizzate sembrano essere il miglior modo per semplificare l'accesso alle risorse di calcolo aggregate in una Grid. La specifica *Web Services Resource Framework* (WSRF), oppure la formalizzazione dei Grid Portal, hanno permesso una standardizzazione dei servizi offerti dalle Grid e della loro modalità di accesso. Questa visione orientata ai servizi, ha portato a pensare alla fruizione del Grid Computing come una *utility on-demand, Utility Computing*, in termini molto simili a quelli formulati nel lontano 1961 da John McCarthy del MIT: "*If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility... The computer utility could become the basis of a new and important industry*".

È ancora importante sottolineare che così come avviene adesso per il termine Cloud, a suo tempo anche per il termine Grid si è avuto un periodo transitorio in cui molte infrastrutture software si dichiaravano essere Grid sebbene avessero poi scopi e strutture assai diversi. Dopo più di un decennio esistono esempi di Grid funzionanti e operative che in modi differenti anticipano aspetti presenti nelle emergenti Cloud. Attualmente esistono Grid puramente orientate all'*High Performance Computing* (HPC) e altre prevalentemente orientate alla sperimentazione software che generalmente da' comunque supporto all'HPC. Queste due tipologie di Grid hanno in comune l'aggregazione di risorse di supercalcolo costituite da *computational cluster* appartenenti a domini amministrativi diversi. Per quanto riguarda il

puro supporto all'HPC, il progetto EGEE [14] - il più imponente per numero di soggetti coinvolti - ha prodotto una Grid che aggrega 41.000 CPU provenienti da circa 150 cluster sparsi su tutto il globo. Tale aggregazione è ottenuta attraverso il *middleware gLiteche* che integra e lega software derivanti dai principali progetti di *middleware* per Grid come quelli provenienti dalla *Globus community* [15], da iniziative europee (*DataGrid*) e intercontinentali (*DataTag*). Simile a EGEE è il progetto *Open Science Grid* che adotta simili strumenti software, sovente sviluppati in cooperazione con EGEE stesso e ne condivide gli scopi. Queste Grid forniscono una piattaforma di calcolo utilizzabile per uno specifico insieme di applicazioni (una ventina, a volte le stesse, sia per EGEE che per *Open Science Grid*) appartenenti a vari domini come la fisica delle alte energie, fluido dinamica computazionale, chimica computazionale, astronomia-astrofisica, bioinformatica ecc.. Una certa applicazione, una volta sviluppata, viene installata su un certo numero di *cluster* con l'aiuto degli amministratori. Entrambe le grid usano dei *software client* customizzati per l'accesso alle risorse di calcolo e consentono la sottomissione di job computazionali con una limitata capacità di configurazione dell'ambiente di esecuzione che viene fornito staticamente dalla grid.
Simile a queste Grid ma orientata a uno sviluppo e una messa in linea delle applicazioni più snella è TeraGrid che aggrega una dozzina di differenti siti di *cluster* computazionali negli Stati Uniti e quindi opera su una scala dimensionale di una decade inferiore a EGEE. Rispetto alle Grid puramente orientate all'HPC, TeraGrid offre significativa capacità di configurazione dell'ambiente di esecuzione attraverso strumenti chiamati *ScienceGateways* che integrano in un portale Web oppure con delle API *Web Services-based* l'accesso per insiemi di strumenti (*e.g. storage*) e applicazioni scientifiche.
Una totale flessibilità di configurazione è perseguita da Grid5000 che aggrega rispettivamente nove siti di super calcolo sparsi sul territorio francese. Infatti, l'obiettivo di Grid5000 è il supporto della sperimentazione di software distribuito su larga scala piuttosto che al puro HPC. Ciò non toglie che tali software, molto spesso, sono orientati all'HPC. Grid5000 adotta un modello basato sulla virtualizzazione dell'hardware. Un utente può configurare un'immagine di un sistema operativo con l'applicazione di interesse e metterla in linea su un insieme di macchine dotate di *hypervisor* previamente riservate ottenendo così un *cluster* di macchine virtuali ritagliato sulle esigenze della desiderata applicazione. L'introduzione della virtualizzazione ovviamente comporta un certo decremento delle prestazioni che comunque può essere tenuto sotto controllo con delle politiche di assegnazione fissa di risorse hardware per macchina virtuale. L'evidente vantaggio è la rapidità di *set-up* per esperimenti che tipicamente richiedono configurazioni complesse e altamente specifiche. L'accesso a Grid5000 avviene attraverso il protocollo SSH attraverso *gateway* specifici presenti in ciascun sito, dai quali poi è possibile operare con strumenti *command-line* per riservare, configurare e accedere alle risorse computazionali.
Le differenze evidenziate riflettono differenti esigenze: il puro supporto ad applicazioni scientifiche e il supporto alla sperimentazione per *distributed computing* su larga scala. In ogni modo, il Cloud Computing indubbiamente abbraccia i concetti evidenziati.
L'idea di aggregare risorse di supercalcolo distribuite geograficamente, trasversale a tutti i progetti di Grid, è evidentemente ripresa dal Cloud Computing sebbene con sfumature diverse. Infatti, non costituisce il target strutturale ma un'eventuale evidenza dettata dalla necessità di offrire un servizio di elevata scalabilità in grado di assorbire richieste da un sempre crescente numero di utenti.
Da progetti come TeraGrid viene ripresa l'idea di adottare potenti interfacce Web con tecnologie provenienti dal Web2.0 e API di facile integrazione software basate su *Web Services* per consentire una configurazione e un accesso semplice e standardizzato a un'infrastruttura di calcolo in realtà complessa. L'utilizzo massiccio della virtualizzazione dell'hardware per dare la massima flessibilità di configurazione dell'ambiente desiderato indubbiamente riporta all'esperienza di progetti come Grid5000.
Con il Cloud Computing, quindi, si tende a dare una grande rilevanza a un'interfaccia semplice e basata su tecnologie web (i.e. *Cloud Interface*), attraverso cui è possibile

Riquadro 3 - Infrastructure In-Cloud

Amazon EC2, abbreviazione di "*Amazon Elastic Compute Cloud*", è senza dubbio l'attuale leader mondiale nel campo del Cloud Computing. L'utente utilizza un browser per le iniziali operazioni di registrazione e *billing* e, successivamente, un insieme di comandi, con interfaccia grafica oppure *command-line*, per riservare *on demand* una certa capacità computazionale in termini di CPU, RAM e storage (http://aws.amazon.com/ec2/#instance) e poi installarvi e configurare molti tipi di Sistemi Operativi (http://aws.amazon.com/ec2/#os), il tutto con un preciso tariffario (http://aws.amazon.com/ec2/#pricing) che va dai 10 centesimi di dollaro all'ora per un'istanza di linux con le capacità più basse a 1,20 dollari per un'istanza di Windows su un *set-up* ad elevate prestazioni. I comandi sono dei programmi Java forniti da Amazon che via Web Services consentono l'interazione con la Cloud. Per dare un'idea della flessibilità di Amazon, basti pensare che sia possibile configurare quali porte aprire su una macchina riservata e poi accedervi "normalmente" via SSH, siccome ciascuna macchina può avere un IP pubblico raggiungibile dall'esterno. Inoltre è possibile integrare una propria applicazione direttamente con la Cloud attraverso l'uso delle API a loro volta utilizzate dai comandi citati in precedenza. Amazon fornisce un insieme di interfacce Web Services basate su WSDL (*Web Service Description Language*) chiamate EC2 WSDLs. La tecnologia su cui si basa il *back-end* della Cloud di Amazon è proprietaria.
Nimbus ed Eucalyptus sono le risposte *open-source* ad Amazon. Entrambi i progetti provengono da gruppi di ricerca già attivi nel Grid Computing ed entrambi offrono un software che consente di creare una Cloud. Nimbus, proveniente dalla *Globus community*, implementa una parte dell'interfaccia EC2 WSDLs e l'interfaccia Grid WSRF e sostanzialmente consente di utilizzare come *back-end* un singolo *cluster* computazionale. Esempi di installazioni operative di Nimbus si trovano su http://workspace.globus.org/clouds/. Eucalyptus, sviluppato nel dipartimento di Computer Science dell'Università della California a Santa Barbara da cui recentemente è derivata un'omonima *spin-off* specializzata in Cloud Computing, consente di aggregare *cluster* multipli come *back-end* ed è compatibile con EC2 WSDLs. Un'installazione operativa di Eucalyptus è accessibile da http://open.eucalyptus.com/wiki/EucalyptusPublicCloud.

Platform In-Cloud

Il concetto di *platfom in-cloud* si avvicina in modo naturale al paradigma di sviluppo delle applicazioni *enterprise oriented* fornito da J2EE o a una dei *frame work* sviluppati per fornire automaticamente la persistenza dei dati di un programma. A fronte di una serie di vincoli più o meno stringenti sulle caratteristiche degli oggetti manipolati dal programma, il *frame work* offre il vantaggio di garantire in modo "trasparente" la persistenza dei dati appoggiandosi ad un meccanismo di *backing storage*. Il parallelo che si ha con le *platform in-cloud* si concretizza nella capacità di queste ultime di garantire, a fronte del pagamento di una tariffa adeguata, il provisioning in automatico di un numero di nodi computazionali e di memorizzazione adeguato a sostenere il livello di carico dell'applicazione. Tra le *platform in-cloud* oggi più note, possiamo citare Google AppEngine, Microsoft Azure e Amazon Elastic MapReduce. Ciascuna di queste vincola in modo più o meno specifico lo sviluppatore. La prima, per esempio, pur fornendo anche un *runtime* dedicato al linguaggio Python, si appoggia sulle caratteristiche di virtualizzazione della Java JVM, sulle Java Persistence API e su Java Data Objects. Azure, al contrario, si appoggia pesantemente su .net, sql server e SOAP. La *platform in-cloud* di Amazon, infine, si basa sul frame *work open source* Hadoop (http://hadoop.apache.org/), lo stesso che viene utilizzato da Yahoo!.

Application In-Cloud

Tra i molti esempi di *application* In-Cloud citiamo SalesForce e Ubikwiti (http://www.ubikwiti.com/) perché rappresentano un interessante risvolto del Cloud e del SaaS. Sono, infatti, applicazioni industriali - rispettivamente un CMR e un ERP - che vengono utilizzate da imprese che sostanzialmente mettono in *outsourcing* completamente una parte molto rilevante del proprio sistema informativo. L'accessibilità via Web, o meglio Web 2.0, consente un *set-up* veramente minimale lato *client* che sostanzialmente necessita solamente di un *browser* delegando a tali *provider* le problematiche relative alla gestione dell'applicazione.

accedere e utilizzare delle risorse computazionali organizzate in una "nuvola" (Cloud) di calcolo. Il termine Cloud è volutamente vago e sottende una tipologia di organizzazione dell'infrastruttura di calcolo slegata teoricamente da qualsiasi vincolo tecnologico. Attualmente, i *back-end* utilizzati per il Cloud Computing impiegano tecniche di virtualizzazione dell'hardware su cluster computazionali singoli (Nimbus [17]), multipli (Eucalyptus [18]) o aggregazioni di centri di calcolo come per la Cloud proprietaria di Amazon che consentono di istanziare dinamicamente uno o più sistemi operativi al di sopra delle risorse fisiche e di organizzare tali risorse in insiemi virtuali omogenei come, ad esempio, cluster computazionali oppure *data center* (riquadro 3). Il Cloud Computing può quindi essere considerato come un'evoluzione del Grid Computing in quanto eredita, mette insieme e rende omogenei in una singola visione concetti, tecnologie e intuizioni sviluppati nel corso degli anni in seno ai più importanti progetti di Grid Computing quali l'aggregazione di centri di supercalcolo, la semplificazione al loro accesso tramite tecnologie web e l'utilizzo di tecniche di virtualizzazione dell'hardware.

4. ...E RIVOLUZIONE

Al di là di aspetti puramente tecnici che, in effetti, accreditano un'ipotesi evoluzionistica o, quantomeno, una diramazione originata dal Grid Computing e, in particolare, dal segmento dell'*Utility Computing*, il Cloud può essere considerato rivoluzionario per quanto riguarda il modello di busi-

ness. Sebbene si basi su uno schema esistente in ambito economico-finanziario, tale schema non è mai stato adottato prima per "vendere" *distributed computing*. Il modello è quello delle *utility* di larga scala dove, data un'infrastruttura in produzione, l'incremento del numero di utenti ha un costo quasi nullo ma moltiplica enormemente i profitti e al contempo contribuisce alla riduzione dei prezzi. Questo modello è universalmente utilizzato per fornire accesso all'elettricità, al gas, all'acqua e alla connettività a rete telefonica e Internet nelle abitazioni degli utenti finali. Il Cloud Computing aggiunge a questa lista CPU *time* e *massive storage*. Amazon, il soggetto che per primo ha commercializzato questa nuova *utility*, offre due tipi di servizio: Compute Cloud EC2 e Data Cloud S3. Il primo consente di prenotare, configurare e istanziare insiemi di macchine per il periodo desiderato. Con il secondo si ha accesso a uno spazio di *storage* praticamente illimitato. L'unico requisito è avere una carta di credito.

Al contrario, il modello di business alla base del Grid *Computing* è *community-based*. Quando un'istituzione decide di unirsi a una Grid cede l'accesso esclusivo alle risorse di calcolo condivise ma, al contempo, acquisisce la possibilità di accedere a un numero di risorse potenzialmente molto più alto. Per esempio TeraGrid riunisce dodici centri di supercalcolo appartenenti ad altrettanti istituti di ricerca universitari in USA. Ai supercalcolatori condivisi hanno accesso tutti i partecipanti che guadagnano una potenziale potenza di calcolo irraggiungibile con un singolo *cluster*. L'accesso esclusivo ceduto a un insieme di utenti più ampio può ottimizzare l'utilizzo delle risorse che non restano più inattive e l'aumentata capacità computazionale consente di affrontare sfide e problemi di complessità altrimenti impossibili. In realtà sono stati proposti modelli di Grid economy per creare un'infrastruttura globale con il supporto del *trading*, della negoziazione, del *provisioning* e dell'allocazione di risorse sulla base del livello di servizio fornito, dei costi e delle richieste degli utenti. Sebbene siano stati anche applicati in pratica modelli economici per lo scambio di risorse, coordinazione di risorse baste sulla teoria dei giochi, valute virtuali e intermediari [19], non c'è ancora stato uno sviluppo significativo della *Grid-economy*. Questo fatto è in realtà legato anche all'ambito, orientato alla ricerca, da cui provengono le Grid che induce una logica *project-based*, la quale stabilisce a priori il livello di coinvolgimento in termini di risorse dei partner in una Grid. Va notato che questo non esclude lo sfruttamento commerciale del Grid computing come testimoniano le attività di aziende come Entropia, Platform e l'italiana Nice che realizzano Grid private, ritagliate sulle esigenze dei propri clienti.

Da questo modello di business viene riflesso un utilizzo senz'altro rivoluzionario ed esclusivamente commerciale delle tecnologie già adottate nel Grid. Infatti, l'utilizzo di tecnologia Web-based congiuntamente alle tecniche di virtualizzazione dell'hardware nel Cloud Computing consente:

1. la fornitura di servizi computazionali con una granularità di tariffazione significativamente piccola;

2. l'adozione di *Service Level Agreements* (SLAs) affidabili anche a fronte della sopra citata granularità di tariffazione.

Nel Grid, l'adozione di queste tecnologie ha significati diversi che non sono legati direttamente allo sfruttamento commerciale e, inoltre, non sono stati armonizzati in una singola visione come nel Cloud. Se da un lato c'è una condivisione di intenti nell'adozione di interfacce basate su standard Web al fine di semplificarne l'accesso, che nel caso del Cloud includono anche il non trascurabile *billing* dell'*utility*, per quanto riguarda la virtualizzazione vi è un'evidente distinzione.

L'intersezione tra Grid e virtualizzazione dell'hardware ha migliorato grandemente la capacità di configurazione offerta agli utenti, ovviamente ove questa è richiesta come nella sperimentazione di *distributed computing* su larga scala. È chiaro, infatti, che l'uso di hardware virtualizzato non porti benefici prestazionali: a parità di hardware disponibile, il livello prestazionale ottenibile da ciascuna di N macchine virtuali è inferiore a 1/N. Questo degrado, per quanto piccolo possa essere, è comunque dovuto alla presenza dell'*overhead* di virtualizzazione e

alla presenza di N istanze di sistema operativo. Quindi, per il Grid Computing, l'adozione della virtualizzazione significa principalmente la possibilità di generare configurazioni di test per lo sviluppo delle applicazioni che poi saranno eseguite sull'hardware reale. Al contrario, in ambito Cloud Computing, l'adozione della virtualizzazione è uno degli aspetti nodali. Questo fatto, quindi, deve sottolineare come esista un aspetto rivoluzionario nell'insieme di applicazioni che Cloud Computing si propone di affrontare. Infatti, se il target di Grid è nelle applicazioni di tipo Grand Challenges, Cloud Computing cancella questa deriva per riportare il fuoco sulle *web-application* vendibili anche loro stesse come servizi e, di fatto, ripristinando la visione di computazione come pubblica utilità che, seppure messa da parte dalle successive evoluzioni dei principali progetti di Grid, è stata la parola chiave del *Grid Book* [7]. Da questo punto di vista, in sintesi, la rivoluzione è forse più assimilabile ad una restaurazione degli obiettivi iniziali, ma rimane tuttavia un cambiamento di fuoco estremamente significativo.

5. NUVOLA O TAPPETO: MARKETING PER NASCONDERE I PROBLEMI IRRISOLTI?

Per quanto il Cloud Computing possieda alcuni aspetti rivoluzionari in termini di *business model*, esistono ancora diversi problemi che, ereditati dal Grid Computing, rimangono sostanzialmente irrisolti e che si pongono come seri ostacoli ad un significativo livello di adozione del Cloud Computing.
Non forniremo qui una lista esaustiva, ma ci limiteremo a citarne due: uno acuito dall'innovativo *business model* proprio del Cloud Computing e uno strettamente ereditato dal Grid Computing.
Il primo, megalitico ostacolo alla diffusione del modello del Cloud Computing è rappresentato dal problema del *downtime*. Tutti ci ricordiamo la definizione di Lamport di un sistema distribuito: "*you know you have one when the crash of a computer you have never heard of stops you from getting any work done*".
Questa può riassumere la prima ragione di difficoltà dei sistemi di Cloud Computing: è estremamente difficile per un'azienda rinunciare completamente al controllo di una componente vitale del suo *core business*.
Seppure questo approccio possa derivare da una reazione non meditata, è pur sempre vero che tutti i principali sistemi di Cloud Computing sono incorsi negli ultimi anni in episodi di interruzione del servizio che, seppure nella maggior parte dei casi non siano stati catastrofici, hanno limitato significativamente l'operatività di chi da loro dipendeva (Figura 3 [21]). Per questa ragione e per il fatto che la reputazione di solidità si costruisce con estrema difficoltà ma si distrugge in un solo sfortunato incidente, l'a-

2008

CCI	Date	Product	Provider	Severity	Incident Type	Incident Sub-Type	Exploit	Affected	Comments
CCI-0006	2008-01-07	Salesforce.com	Salesforce.com	High	Outage	Network Outage	No	All	Affected all instances and supporting infrastructure
CCI-0005	2008-10-16	AWS Services	AWS	High	Security	Man-in-the-Middle	No	All	Issue present since service launch
CCI-0004	2008-10-15	Gmail	Google	High	Outage	502 error	No	Unknown number of users	Lasted more than 24 hours
CCI-0002	2008-09-18	Google Docs	Google	High	Security	Session Hijacking	No	Some Thai Users	Limited to ISP(s) in Thailand
TBA	2008-09-15	App Engine	Google	Low	Outage	Performance Degradation	No	All	Datastore writes experienced elevated latencies and error-rates.
TBA	2008-09-02	Google Apps	Google	High	Security	User Impersonation	Yes [11]	All SSO users	Malicious service provider could impersonate a user at other service providers.
TBA	2008-08-26	FlexiScale	FlexiScale	Critical	Outage	Disaster Recovery	No	All	Full extended outage
TBA	2008-08-12	Gmail	Google	High	Outage	Change Management	No	Many	Users unable to use webmail due to issues with loading contacts between 14:00 and 16:00 PT
TBA	2008-08-08	The Linkup	Nirvanix MediaMax	Critical	Data Loss	Closure	No	20,000	Data claimed to be safe but inaccessible
TBA	2008-07-20	Amazon S3	AWS	Critical	Outage	Design Fault	No	All	Full outage for 8 (weekend) hours
CCI-0003	2008-07-10	MobileMe	Apple	Moderate	Outage	Migration	No	All	Scheduled outage window exceeded during upgrade to MobileMe
CCI-0003	2008-07-09	Mac	Apple	Info	Outage	Scheduled Outage	No	All	Full outage (except mail) during upgrade to MobileMe 18:00-00:00
TBA	2008-04-28	EC2	Amazon	Low	Outage	Degraded Performance	No	Small subset of instances	Result of a customer creating a large number of firewall rules and instances.
TBA	2008-02-15	Amazon S3	AWS	Low	Outage	Authentication Failures	No	All	Early morning outage (04:31-06:48 PST) caused by authentication service overload

FIGURA 3
Incidenti causa di downtime per sistemi Cloud nel 2008

dozione massiccia di sistemi di Cloud Computing per i servizi vitali di un'azienda è ancora piuttosto difficile.
Un secondo problema irrisolto per il Cloud Computing è quello del modello di programmazione. Seppure sia vero che Cloud Computing è stato spesso associato alle *web applications* e quindi ad un modello di programmazione di non estrema complessità, è altrettanto vero che questo modello (sia esso Map Reduce, sia esso Farmer Workers) si basa sull'intrinseca mancanza di comunicazione tra i *task* paralleli. Questa caratteristica, se anche è facilmente riscontrabile nella maggior parte dei processi di ricerca di *pattern* all'interno di un *data set* di grandi dimensioni, non è generalizzabile a tutte le applicazioni e rimanda quindi allo stesso problema con cui si è precedentemente scontrato il Grid Computing: la programmazione parallela è difficile.

6. CONCLUSIONI

Se si guarda oltre la nube (gioco di parole intenzionale) di marketing che oggi esalta il Cloud Computing (The Economist, nell'Ottobre 2008 scrive un articolo che descrive l'avvento del Cloud Computing con le stesse enfatiche parole usate alcuni anni orsono per il *WorldWideWeb*), un aspetto rivoluzionario del Cloud Computing può essere trovato nel modello di business da esso implicato. La capacità di fornire un ambiente di esecuzione che sia:
1. ritagliato su misura della vostra applicazione;
2. isolato e protetto da interferenze di altre applicazioni;
3. disponibile a richiesta in tempi molto rapidi;
4. tariffato con granularità fine (e.g. ad ore);
rappresenta effettivamente un passo significativo verso l'idea di computazione come servizio di pubblica utilità preconizzato nel *blue book* del Grid Computing [7] ed ancor prima da Leonard Kleinrock [22]. Nonostante questo, esiste ancora un insieme di problemi irrisolti, semplicemente ereditati dal Grid Computing, che limitano, ad oggi, l'uso del Cloud Computing e lo tengono ancora a significativa distanza dal traguardo di diventare la quinta *public utility*.

Bibliografia

[1] Lin G., Dasmalchi G., Zhu J.: *Cloud Computing and IT as a Service: Opportunities and Challenges*. Proc. of the IEEE International Conference on Web Services, 23-26 Sept. 2008.

[2] Erdogmus H.: Cloud Computing: Does Nirvana Hide behind the Nebula?, *IEEE Software*, Vol. 26, Issue 2, March-April 2009 p. 4-6.

[3] Aa.vv.: *Twenty Experts Define Cloud Computing*. SYS-CON Media Inc, http://cloudcomputing.sys-con.com/read/612375_p.htm, 2008

[4] Aa.vv.: *The Open Grid Forum*. http://www.ogf.org/

[5] Aa.vv.: *Enter the Grid website*. http://enterthegrid.com/

[6] Gentzsch W.: *Grids are dead! Or are they?* On Demand Enterprise feature article, June 2008, http://www.on-demandenterprise.com/features/26060699.html?viewAll=y

[7] Foster I., Kesselman C.: *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, San Francisco, 1999.

[8] Grossman R.L.: The Case for Cloud Computing. *IT Professional*, Vol. 11, Issue 2, March-April 2009, p. 23-27.

[9] Dean J., Ghemawat S.: MapReduce: *Simplified Data Processing on Large Clusters*. Proc. Of Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, December, 2004.

[10] Armbrust M., Fox A., Griffith R., Joseph A.D., Katz R., Konwinski A., Lee G., Patterson D., Rabkin A., Stoica I., Zaharia M.: *Above the Clouds: A Berkeley View of Cloud Computing*. www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf

[11] Aa.vv.: *Sito web di Google AppEngine*. http://code.google.com/appengine/

[12] Aa.vv.: *Sito web di Microsoft Azure*. http://www.microsoft.com/azure/windowsazure.mspx

[13] Aa.vv.: *Sito web di Amazon Elastic MapReduce*. http://aws.amazon.com/elasticmapreduce/

[14] Aa.vv.: *The EGEE2 Project*. http://egee2.eu-egee.org/

[15] Aa.vv.: *The Globus Project*. http://www.globus.org/

[16] Aa.vv.: *Nimbus*. http://workspace.globus.org/

[17] Aa.vv.: *Eucalyptus Systems*. http://www.eucalyptus.com/

[18] Buyya R., Yeo C.S., Venugopal S., Broberg J., Brandic I.: Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5-th utility. *Future Generation Computer Systems*, Vol. 25, 2009, p. 599-616.

[19] Aa.vv.: *Amazon Elastic Compute Cloud*. http://aws.amazon.com/ec2/

[20] Aa.vv.: *Cloud Computing: Incidents Database*. http://wiki.cloudcommunity.org/wiki/CloudComputing:Incidents_Database

[21] Kleinrock L.: A vision for the Internet. S*T Journal of Research*, Vol. 2, n. 1, 2005, p. 4-5.

[22] Foster I., Iamnitchi A.: *On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing*. Proc. of the 2-nd International Workshop on Peer-to-Peer Systems, 20-21 February 2003, Berkeley, CA, USA.

[23] Ganguly A., Wolinksky D.I., Boykin P.O., Figueiredo R.: *Improving Peer Connectivity in Wide-area Overlays of Virtual Workstations*. Proc. of the 7-th International Symposium on High Performance Distributed Computing, 23-27 June 2008, Boston, MA, USA.

Mauro Migliardi è nato a Genova nel 1966. Dopo essere stato uno dei principali ricercatori del progetto HARNESS per il meta e Grid computing presso la Emory University di Atlanta, è stato ricercatore universitario presso l'Università di Genova ed è ora Professore associato presso l'Università di Padova. Mauro Migliardi ha pubblicato oltre ottanta articoli scientifici soggetti a peer-review e ha tra i suoi principali interessi di ricerca le tecnologie e le metodologie per la progettazione e lo sviluppo di sistemi software distribuiti complessi.
E-mail: mauro.migliardi@unipd.it

Roberto Podestà si è laureato in Ingegneria Informatica nel 2003 e ha conseguito il dottorato in Scienze e Tecnologie dell'Informazione nel 2007 presso l'Università di Genova. I suoi interessi scientifici riguardano le architetture software distribuite e le tecniche di programmazione concorrente e parallela. Dopo una breve esperienza in Australia verso la fine del dottorato presso il GridsLab dell'università di Melbourne, è tornato a Genova collaborando in un progetto congiunto tra l'università e Telecom Italia. Nel settembre 2007, ha vinto un post-doc bandito dall'ERCIM (*European Reseach Institute for Informatics and Mathematics*) e si è quindi trasferito all'INRIA (*Institut National de Recherche en Informatique et en Automatique*) nella sezione a sud di Parigi. L'anno successivo è stato confermato all'INRIA con una posizione da expert engineer. La sua attività si concentra sulla progettazione di middleware all'interno del principale progetto francese di cloud computing.
E-mail: ropode@gmail.com