CARACTERISTICI PARADIGMATICE, TEHNOLOGICE ŞI ECONOMICE ALE HPC, GRID, HSC, CLOUD ŞI VOLUNTEER COMPUTING – DELIMITATOARE ŞI DETERMINANTE ALE VOLUNTEER CLOUD COMPUTING

Bogdan Enciu

Institutul Național de Cercetare-Dezvoltare în Informatică – ICI Bucureşti

enciu@ici.ro

"*Paralelismul constituie regula,*
secvențialitatea constituie excepția inevitabilă."
T. Muntean

Rezumat: Articolul prezintă caracteristici, de diverse naturi, specifice tipurilor de tehnologii de calcul menționate. Sunt descrise, la un nivel general, aspecte definitorii ale acestor tehnologii şi sunt evidențiate diferențele dintre acestea. Descrierea relevează şi evoluția dezvoltării tehnologiilor şi legăturile dintre ele. De asemenea, prezentarea acestor caracteristici urmăreşte şi lămurirea unor aspecte care pot genera confuzii, în special, determinate de diferitele posibilități de utilizare ale tehnologiilor, acesta fiind principalul motiv pentru care s-a intenționat demararea acestui demers după încheierea proiectului de dezvoltare a 'site'-ului Grid din ICI Bucureşti, demers realizat acum. Este descrisă problematica virtualizării, cu accent pe calculul voluntar, aceasta constituind un aspect determinant al evoluției acestei tehnologii de calcul către Volunteer Cloud.

Cuvinte cheie: HPC, Grid, HSC, Cloud, Volunteer Computing, Volunteer Cloud, virtualizare.

Abstract: This paper presents characteristics, of various natures, specific for the types of computing technologies mentioned. Are described, at a general level, defining aspects of these technologies and are highlighted the differences between them. Description reveals the evolution of the development of the technologies and the inter-relations between them. Presenting these features, it also aims clarifing certain aspects that could create confusion, especially due to the different possible uses of these technologies, this being the main reason of the intention for launching this action after the end of the project regarding the development of the Grid site in ICI Bucharest, now realized step. The paper describes the issue virtualization, focusing on volunteer computing, this being a crucial aspect of the evolving of this computing technology towards Volunteer Cloud.

Keywords: HPC, Grid, HSC, Cloud, Volunteer Computing, Volunteer Cloud, virtualization.

1. Tehnologiile Grid, Cloud, Volunteer Computing – definiții şi deziderate

În contextul în care a avut loc o continuă dezvoltare atât a tehnologiilor de rețea cât şi a cerințelor în acest sens, a apărut ca naturală şi cerința de adaptare a implementărilor bazate pe calculul de înaltă performanță de la tehnologia 'mainframe'-urilor la cea a 'cluster'-elor interconectate şi, ulterior, la cea a scalabilității 'cluster'-elor. Această tendință, bazată şi pe dezvoltarea capacității de calcul a sistemelor personale, a condus orientarea rezolvării unor probleme complexe de calcul de la tehnologia bazată pe supercalculatoare la tehnologia calculului voluntar.

Tehnologia Grid implică partajarea coordonată a resurselor şi rezolvarea de probleme în cadrul unor organizații virtuale multi-instituționale şi dinamice [9]. Inițial, resursele erau doar fişiere dar, ulterior, resursele au fost extinse până la accesul direct al calculatoarelor, datelor, programelor şi al altor tipuri de resurse. Tehnologia Grid furnizează o infrastructură globală de 'cluster'-e interconectate constând din resurse de calcul dispersate în cadrul Internet. Dezideratul calculului Grid a fost acela de a transforma calculul într-o utilitate obişnuită, precum energia electrică, gazele, apa. Calculul Grid ridică probleme privind securitatea, eterogenitatea şi portabilitatea.

Calculul Cloud este o paradigmă considerată ca marcantă în calcul şi poate fi definit în mai multe moduri. Este considerat ca un model pentru permiterea accesului în rețea pretutindeni, în

mod convenabil, la cerere, la o masă de resurse de calcul configurabile (rețele, 'server'-e, unități de stocare, aplicații şi servicii) care pot fi rapid aprovizionate şi furnizate cu minim efort de gestionare sau interacțiune cu furnizorul de servicii [3]. Altă definiție este aceea că acest tip de calcul se referă atât la aplicațiile furnizate ca servicii în cadrul Internet cât şi la 'hardware'-ul şi 'software'-ul de sistem din centrele de date care furnizează acele servicii. Cloud are o mulțime de caracteristici prin care se pot aborda probleme de calcul. De asemenea, Cloud este definit [3] ca o formă de calcul Grid în care resursele virtuale sunt în mod dinamic alocate. În definiția NIST (the US National Institute of Standards and Technology) [11] se referă la: cinci caracteristici (auto-service la cerere, acces în rețea de bandă largă, interogare de resurse, elasticitate rapidă, serviciu măsurat), patru modele de utilizare (cloud privat, în care datele şi procesele sunt gestionate din interiorul unei organizații; cloud comunitar, în care infrastructura este partajată între câteva organizații; cloud public, în care infrastructura este disponibilă public şi poate fi accesată prin 'Web'; cloud hibrid, o combinație a primelor două, cu scopul portabilității cu tehnologia standardizată), trei modele de servicii ('software'-ul ca serviciu – SasS), platforma ca serviciu – PasS, infrastructura ca serviciu – IasS).

Calculul voluntar (Volunteer Computing) se referă la înrolarea utilizatorilor în sensul de a permite partajarea resurselor neutilizate ale calculatoarelor lor, care sunt conectate la o rețea, pentru rezolvarea unor probleme computaționale mari. Procesul de bază se derulează astfel: un 'client' rulează pe o maşină oferită voluntar şi, din timp în timp, contactează 'server'-ul proiectului pentru a-i întoarce rezultatele calculate şi 'job'-urile solicitate. Acest model este în principiu utilizat pentru 'job'-uri care folosesc intensiv unitățile centrale de prelucrare ale calculatoarelor şi care pot fi împărțite în 'task'-uri mai mici ce pot fi executate de gazde diferite [8]. Cel mai important aspect îl constituie proiectarea într-un mod facil şi transparent al acestei idei. Există multe proiecte de Volunteer Computing (unul dintre cele mai populare fiind SETI@home, început în 1999 şi care a atins în 2014 peste trei milioane de voluntari). Combinarea calculului Cloud cu calculul Volunteer a condus la crearea unei noi paradigme de calcul, Volunteer Cloud Computing [3] [8].

În acest context, pentru unii utilizatori, ca şi pentru cei care intenționează să devină voluntari, pot apărea, şi apar, interpretări şi chiar confuzii cu privire la aceste tehnologii, la evoluția lor şi la posibilitățile de utilizare a lor. Pot apărea întrebări precum: Tehnologia Grid este precursoare a tehnologiei Cloud sau acestea sunt doar tehnologii tangențiale ? Calculul de înaltă performanță (High Performance Computing - HPC) şi Grid desemnează acelaşi lucru ? Cloud, dacă poate suporta încărcări de lucru de tip HPC, este HPC sau este HSC (High Scalability Computing) ? Se poate vorbi de o convergență între HPC şi Cloud Computing ? Volunteer Computing este calcul Grid sau calcul Cloud ? Sau Volunteer Computing este o altă formă de tehnologie de calcul distribuit alături de Grid, Cloud ? Dar Volunteer Cloud ... ?

2. Caracteristici – indicatoare de ortogonalitate, tangențialitate, similaritate, convergență

2.1 Calcul paralel şi calcul distribuit

Calculul paralel şi calculul distribuit sunt expresia corelării între algoritmii care identifică şi exploatează paralelismul şi caracteristicile funcționale ale arhitecturii unui sistem de calcul sau a unei structuri de sisteme de calcul.

Calculul paralel real implică utilizarea de maşini (masiv) paralele, aşa-numite sisteme cu interconectare puternică. Calculul distribuit se bazează pe un ansamblu de elemente independente de prelucrare, distribuite spațial, care sunt interconectate şi care cooperează pentru realizarea sarcinilor de calcul.

Mediile paralele/distribuite necesită utilizarea unor modele de programare. Programarea secvențială necesită simularea mediilor existente pe sisteme paralele (sisteme de operare, instrumente etc.) şi exploatarea automată a paralelismului. Programarea concurentă sau quasi-paralelă se referă la potențialitatea execuției simultane, nu la o execuție realmente

simultană/paralelă. O altă formă de programare implică utilizarea explicită a paralelismului real, adesea pentru respectarea restricțiilor topologice dintr-o rețea fixă (utilizatorii manipulează implementarea paralelă a problemei, în sensul că, cea mai mare parte a codului poate fi făcută într-un limbaj standard dar cu noi elemente necesare pentru gestiunea comunicării şi a concurenței). La nivelul paralelismului "fin" se utilizează modele sofisticate ale concurenței şi paralelismului. Structura sistemului de calcul sau a sistemelor de calcul interconectate dictează utilizarea specifică a acestor modele. Oricum, regula o constituie paralelismul, în timp ce secvențialitatea constituie excepția inevitabilă [12].

Interacțiunile dintre procese/'task'-uri/'thread'-uri (văzute ca entități de programare independente şi unități de distribuire şi execuție cu context propriu) sunt una dintre, sau combinații ale, următoarele/lor: cooperare (pentru realizarea unei sarcini/serviciu), concurență (ca rezultat al accesului mai multor procese la o resursă), comunicare (între procesele cooperante sau pentru sincronizarea accesului concurent la aceeaşi resursă). Cunoaşterea/ estimarea nivelului comunicațiilor dintre procese este un aspect important în stabilirea strategiilor de planificare la execuție care să evite apariția fenomenului de saturație.

2.2 Performanță şi scalabilitate

Între performanță şi scalabilitate este o diferență fundamentală deşi adesea aceste concepte ajung să fie considerate oarecum sinonime [6]. Performanța constituie capacitatea unei componente 'hardware-software' de a oferi o anumită cantitate de "ieşire". Scalabilitatea exprimă abilitatea de extensie a unui sistem pentru a corespunde cerințelor. Aceasta este măsurată prin observarea performanței combinate/sinergice a componentelor individuale ale sistemului şi cum acestea funcționează de-a lungul timpului. Cu alte cuvinte, performanța măsoară capacitatea unei singure componente din cadrul unui sistem, pe când scalabilitatea măsoară abilitatea de extensie a unui sistem mare.

Sistemele scalabile pot avea componente care au o performanță relativ scăzută dar care să permită adăugarea oricâtor componente.

Sistemele de înaltă performanță urmăresc mai degrabă să exploateze la maxim fiecare resursă a fiecărei componente şi nu să se focalizeze asupra întregii structuri. Pot exista sisteme de înaltă performanță într-un sistem foarte scalabil sau nu.

Pentru majoritatea scopurilor, performanța şi scalabilitatea sunt concepte ortogonale dar, cu toate acestea, uneori sunt considerate chiar echivalente sau în relație de interdependență.

2.3 Calcul HPC şi Grid

Originile tehnologiilor de calcul HPC şi Grid se află în organizațiile de cercetare unde cerințele privind prelucrarea unor mari cantități de informații au crescut continuu (fiind vorba de date provenite din fizica nucleară, de la sateliți, de la genomuri etc.). Grid-ul există de la începutul erei calculului instituțional când a devenit mai uşor pentru instituțiile de cercetare să se orienteze de la super-calculatoarele de tip 'mainframe' către un model de dezvoltare prin scalare utilizând o multitudine de maşini x86 relativ ieftine organizate în 'cluster'-e mari [6].

Majoritatea 'cluster'-elor x86 au fost construite pentru foarte mare performanță şi scalabilitate (dar cu accent asupra performanței componentelor individuale, 'server'-e şi rețeaua de interconectare). Raportul dintre preț şi performanță al sistemului în ansamblu nu a fost la fel de important ca ieşirea combinată a întregului sistem. În majoritatea organizațiilor de cercetare în care s-au construit rețele grid (căutându-se să se achiziționeze cel mai performant 'hardware' cu bugetul disponibil) se încearcă să se extragă maximum de performanță din fiecare componentă [6].

2.4 Calcul HSC şi Cloud

HSC şi Cloud Computing se concentrează asupra atingerii unui raport cât mai bun între preț şi performanță, utilizând sisteme de preț scăzut, cu componente realizate pe baza unor standarde

deschise (diferențele între diferiți producători fiind reduse) şi achiziționând multe de astfel de sisteme/componente. Aceasta înseamnă construirea de sisteme foarte mari şi scalabile. Numărul de unități centrale de prelucrare ('core'-uri) a ajuns de ordinul $10^6 \div 10^7$. Amazon EC2 a depăşit milionul de 'core'-uri (Amazon Elastic Compute Cloud este un serviciu 'web' care furnizează capacitate de calcul reajustabilă în cloud, fiind proiectat să facă calculul Cloud mai facil pentru dezvoltatori [3]). De asemenea, Google are de ordinul zecilor de milioane de 'core'-uri [6]. Cloud-urile construite pe principiul calculului de mare scalabilitate sunt de un ordin de mărime mai mari decât majoritatea 'cluster'-elor Grid şi sunt proiectate să manipuleze încărcări de lucru generice cerând atingerea celui mai bun raport preț/performanță atunci când sunt construite.

2.5 Încărcările de lucru Grid şi Cloud

Încărcările de lucru Grid şi Cloud pot fi foarte diferite.

Există două tipuri principale de probleme care sunt rezolvate pe 'cluster'-e de tip Grid: probleme care implică transfer de mesaje MPI (Message Passing Interface) şi probleme cu un caracter paralel pentru care nu se impune un efort pentru a partaja problema în 'task'-uri paralele (aşa-numite Embarrassingly Parallel Problems - EPP) [6] [10].

MPI este principalul model pentru programarea HPC, fiind cel mai potrivit pentru rezolvarea majorității problemelor computaționale complexe [10]. MPI este o paradigmă de programare care permite preluarea de mulțimi extrem de mari de date şi analiza informației în paralel, partajându-se datele între nodurile de calcul. Uneori această prelucrare este denumită şi 'clustering'. Anumite tipuri de probleme necesită această partajare deoarece rezultatele calculate pe un nod pot afecta rezultatele calculate pe alt nod în rețea. Rețelele bazate pe MPI, standardul pentru majoritatea organizațiilor de cercetare, sunt construite pentru maximum de ieşire şi performanță per sistem, inclusiv cea mai mică latență posibilă. Majoritatea utilizează tehnologia Infiniband pentru a transforma întreaga rețea într-un unic supercalculator. Un 'cluster' (Grid) MPI arată sub multiple aspecte mai mult ca un 'mainframe' şi tehnologii precum Infiniband transformă rețeaua într-o magistrală de mare viteză, exact ca o magistrală PCI dintr-un 'server' de tip x86.

Tehnica MPI apare ca restricționată la un singur 'cluster', în sensul că nu se pot transfera mesaje prin Internet pentru a utiliza resurse de calcul ale unor 'cluster'-e distincte. Dar s-au propus modele care oferă posibilitatea de transfer de mesaje între aplicații paralele prin intermediul Internet-ului (un model este bazat pe arhitectura A-JUMP – Architecture for Java Universal Message Passing şi magistrala ESB - Enterprise Service Bus numită şi HPC Bus [10]).

Încărcările de lucru EPP nu necesită partajări. O mulțime foarte mare de date este spartă în bucăți, distribuite la un număr mare de unități de prelucrare şi ulterior datele sunt aduse înapoi şi reasamblate. Aceasta este funcționalitatea pentru MapReduce a lui Google şi a instrumentului 'open source' Hadoop [6]. Încărcări de lucru EPP sunt rulate în mod obişnuit pe 'cluster'-e MPI, deşi unele organizații de cercetare construiesc grid-uri separate sau mai mici pentru a le rula.

Aceste două tipuri de încărcări de lucru, MPI şi EPP, au necesități diferite. Se consideră că majoritatea încărcărilor de lucru Grid sunt de tip EPP [6].

Manipulând încărcări de lucru generice, caracteristicile încărcărilor de lucru Cloud sunt oarecum confuze deşi acest aspect este critic pentru proiectarea de noi 'job'-uri Cloud şi noi sisteme de gestiune a resurselor. S-au efectuat studii pentru caracterizarea comprehensivă a încărcărilor în centre de date reale şi s-au pus în evidență diferențe între diferite centre de date de tip Cloud şi Grid, atât din perspectiva încărcărilor de lucru (job'-uri şi 'task'-uri) cât şi a maşinilor. Au fost analizate lungimea 'job'-urilor, frecvența de transmitere a 'job'-urilor şi utilizarea resurselor în diferite sisteme. De asemenea, s-au efectuat statistici cu privire la încărcările maxime ale maşinilor, stările cozilor de aşteptare şi nivelurile de utilizare relative, cu diferite priorități ale 'job'-urilor şi atribute ale maşinilor. S-a constatat că centrul de date Google indică o alocare mai fină a resurselor în ceea ce priveşte unitatea centrală de prelucrare şi memoria decât în cazul sistemelor Grid/HPC şi, de asemenea, că 'job'-urile sunt transmise cu o frecvență mult mai mare şi sunt mult mai scurte decât 'job'-urile Grid [14].

Se consideră că HSC este ideal pentru majoritatea datelor de lucru de tip EPP. De exemplu, în legătură cu acest tip de activitate, cantităţi mari de sarcini în forma ‘job’-urilor MapReduce au rulat pe Amazon EC2 de la începuturi şi au determinat mare parte din creşterea sa. HPC este diferit deoarece multe încărcări de lucru MPI necesită latenţă mică şi ‘server’-e de mare performanţă. Oricum, nu toate încărcările MPI sunt aceleaşi. Există încărcări MPI care necesită o reţea puternică dar nu o reţea Infiniband. Pentru menţinerea tendinţei serviciului AWS (Amazon Web Service) de a se dezvolta prin utilizarea tehnicilor Cloud, HPC-ul oferit nu utiliza Infiniband ci fibră optică pentru Ethernet de 10G. Aceasta a făcut ca reţeaua să fie puternică, deşi nu extraordinară, şi permite crearea unui serviciu cloud potrivit pentru multe ‘job’-uri HPC. Unele ‘benchmark’-uri arătau că sistemul HPC Cloud al AWS are o performanţă deosebită pentru multe încărcări de lucru de tip MPI [6].

2.6 Volunteer Computing

În cadrul calculului voluntar, pentru a rezolva probleme ştiinţifice este necesar a stabili un anumit cadru de lucru. Scopul principal este acela de a “secţiona” sarcina şi de a o distribui peste tot în lume şi de a primi rezultatele înapoi. Nu există comunicaţie între ‘job’-uri, deci programul trebuie sa permită paralelizarea. Există mai multe cadre de lucru pentru Volunteer Computing, cel mai bine cunoscut fiind ‘middleware’-ul BOINC (Berkeley Open Infrastructure for Network Computing) care are o abordare ‘client-server’ în care ‘client’-ii sunt responsabili de realizarea ‘job’-urilor şi ‘server’-ul are rol de gestionare şi coordonare în sistem [3].

Proiectele Volunteer Computing încearcă să rezolve probleme presante legate de sănătate, mediu sau experimente fizice. Voluntarii sunt interesaţi să contribuie la aceste proiecte în principal datorită subiectului problemelor pe care proiectele sunt interesate să le rezolve.

Utilizat fiind începând din anul 2002, ‘middleware’-ul BOINC a fost, de asemenea, adoptat de către cea mai mare reţea de calcul publică: World Community Grid. Există milioane de utilizatori BOINC înregistraţi şi sunt conectate un număr de calculatoare gazdă care se poate compara cu numărul de ‘server’-e al unor companii gigant, precum Amazon, Google, eBay, Yahoo, Microsoft, IBM, HP/EDS, GoDaddy [8], având în anul 2012 o capacitate operaţională în jur de 7,5 petaflops.

Există însă şi aspecte care sunt considerate, dacă nu negative, cel puţin slabe în legătură cu Volunteer Computing: statutul anonim al voluntarilor, problemele ridicate de insecuritatea fiecărui calculator, de rea-intenţia unor utilizatori care întorc rezultate greşite cu un anumit scop, faptul că încărcările de lucru trebuie să fie tolerante la erori/greşeli deoarece “căderea” unei maşini nu este chiar aşa de rară şi faptul că încărcările de lucru trebuie să aibă rată mică de I/E deoarece voluntarii nu oferă garanţii cu privire la lărgimea de bandă. Toate aceste aspecte nu pot fi pe deplin controlate aşa cum sunt într-un ‘cluster’ de ‘server’-e omogene. Cu toate acestea, având în vedere că există câteva sute de dezvoltatori şi testeri voluntari în comunitatea BOINC, se poate considera acest ‘middleware’ ca o platformă robustă de ‘software’ ‘open-source’ [8].

2.7 Costuri şi eficienţă

Se impune adesea să se evalueze costurile utilizării resurselor distribuite. Costurile pot fi împărţite în costuri privind dezvoltarea (achiziţii de ‘hardware’ şi ‘software’) şi costuri legate de operarea zilnică (energie electrică, personal etc.) [2].

HPC necesită ‘software’ specializat şi dezvoltare ‘hardware’, implicând suport pentru construirea şi gestionarea unui super-calculator (în general, a existat suport din partea furnizorilor). Super-calculatoarele au constituit şi constituie încă vârful în privinţa capacităţii de calcul ceea ce determină costuri zilnice ridicate. Pe de altă parte, deoarece super-calculatoarele utilizează o putere maximă de calcul pentru a rezolva probleme complexe, timpii de execuţie sunt în general reduşi şi astfel utilizarea acestor super-calculatoare este necesară pentru perioade limitate.

Calculul Grid poate fi o cale mai economică de a atinge capacităţi de procesare HPC. Dezvoltarea poate fi un proces lung deoarece se lucrează cu sisteme diferite şi nu se poate impune ce să ruleze o anume organizaţie pe maşinile ei. Oricum, calculul Grid lucrează pe un sistem liber

la momentul utilizării deci rularea unei analize de îndată ce există aplicația ar trebui să nu aibă cost pentru un cercetător individual. Organizația ar trebui să plătească un operator să gestioneze şi să întreţină sistemele grid crescând astfel costurile de rulare.

Pentru calculul Cloud costurile de dezvoltare sunt de obicei mici deoarece cercetătorii nu trebuie să dețină infrastructura fizică. Prețul de lansare a aplicațiilor în cloud pot fi mai scăzute datorită 'hardware'-ului mai ieftin şi utilizării mai eficiente a resurselor. Utilizatorii trebuie să plătească pentru fiecare utilizare ceea ce poate determina costuri potențial ridicate pentru utilizări zilnice (acest mod de utilizare este denumit 'utility computing', similar cu plata pentru o utilitate publică).

Pentru Volunteer Computing, costurile de operare sunt mai mici decât pentru Cloud deoarece puterea de prelucrare este oferită gratis de către public. Oricum, costurile pentru dezvoltări pot fi mai mari deoarece aplicația trebuie să fie adaptată pentru Volunteer Computing. În cazul Cloud Computing teoretic aplicația există deja fiind necesar doar de a "spune" cloud-ului ce anume se doreşte. În ultimii ani s-a constatat o creştere a numărului proiectelor de clacul voluntar, ceea ce reprezintă o recunoaştere crescândă a potențialului calculului voluntar ca abordare eficientă din punctul de vedere al costului pentru gestiunea sarcinilor intensiv computaționale. Există două tipuri de aspecte referitoare la Volunteer Computing: aspectele computaționale (legate de alocarea şi gestiunea 'job'-urilor de calcul) şi aspectele participaționale (legate de atragerea unui număr cât mai mare de voluntari care să îşi ofere resursele de calcul pentru un proiect) [5]. Oricum, în timp ce aspectele computaționale captează atenția cercetătorilor, aspectele participaționale au rămas aproape neexplorate [1].

2.8 Securitate şi acces

Un aspect important se referă la încrederea dintre furnizori de resurse şi utilizatori, în special când nu se cunosc [2]. Partajarea resurselor şi politicile de securitate sunt aspecte conflictuale în centrele de calcul individuale şi pe calculatoare individuale, securitatea fiind astfel un aspect esențial. Toate instrumentele de calcul distribuit au mecanisme de gestiune a securității. Grid utilizează certificate. Furnizorii Cloud au adoptat diverse soluții, precum criptografia. Pentru Volunteer Computing s-au dezvoltat noi soluții, inclusiv tehnici complicate de criptare a datelor.

Accesul eficient la resursele de calcul este, de asemenea, important [2]. Tehnologiile Cloud au fost dezvoltate pentru a permite închirierea rentabilă a infrastructurii de calcul şi de a furniza servicii la cerere. Serviciile Grid au fost dezvoltate pentru a partaja resurse între deținători sau organizații. În cazul HPC, o singură organizație utilizează puterea de calcul a unui singur super-calculator. Volunteer Computing utilizează resurse deținute de către public.

Sub aspectul accesului şi furnizării, HPC, Grid şi Cloud sunt în general simetrice: o organizație poate furniza şi accesa simultan resurse. Acest aspect este relevant pentru Grid (accesul are loc la propriile resurse). Volunteer Computing este asimetric deoarece voluntarii furnizează resursele dar nu primesc nimic în schimb [2].

În cazul HPC, Grid şi Cloud, utilizatorul trimite ('push') un 'job' către resurse. În cazul Volunteer Computing, calculatoarele solicită ('pull') de lucru de la 'server'.

În cazul HPC şi Grid trebuie lucrat cu 'software' standard bine definit dar se pot solicita actualizări. În cazul Cloud exista posibilitatea de a defini şi instala aplicații, sisteme de operare şi alt 'software' necesar. În Volunteer Computing, deoarece resursele sunt oferite de public, acestea sunt definite de cerințele lor şi nu de către cercetători [2].

3. Virtualizarea – concepte şi mecanisme

Virtualizarea oferă o interfață diferită (şi nu neapărat una mai simplă). Conceptual, abordările în sensul virtualizării s-ar putea clasifica în abordarea tip stivă, cu o serie de niveluri abstracte, şi abordarea prin două dimensiuni, verticală şi orizontală. Dar nici una dintre aceste taxonomii nu se poate numi standard deoarece există mai multe moduri de clasificare [13]. Virtualizarea poate fi

aplicată la diverse tipuri de resurse (unități de calcul, unități de stocare, rețele). Evident, există diferite niveluri la care se poate aplica virtualizarea (nivel sistem, nivel proces, nivel sistem de operare) [13].

La nivelul sistem, virtualizarea constă în emularea unui calculator similar cu unul real, fizic (cu UCP, memorie, discuri, interfață de rețea). Se creează o maşină virtuală care acționează ca un calculator real cu un sistem de operare complet. Se disting virtualizarea completă (simulare aproape totală), virtualizare parțială (nu este simulat întreg mediul, astfel încât unele programe pot necesita modificări pentru a rula în mediul virtual), paravirtualizare (nu este simulat mediul 'hardware' iar programele pot rula în domeniile lor izolate dar trebuie să fie modificate în mod specific pentru a rula astfel – se prezintă o interfață 'software' maşinilor virtuale care sunt similare dar nu identice cu aceea a 'hardware'-lui de bază, intenția fiind aceea de a reduce timpii de execuție pierduți pentru operații care sunt substanțial mai dificile de rulat într-un mediu virtual în comparație cu un mediu non-virtual).

La nivel proces, maşina virtuală rulează o aplicație/un proces. Aplicația trebuie să fie scrisă în mod specific pentru maşina virtuală. De obicei se implementează deasupra unui sistem de operare. Are ca avantaj faptul că aplicația este portabilă pe platformele care suportă maşina virtuală (de exemplu Java Virtual Machine pe Windows, Linux, PDA-uri etc.). Are dezavantajul că toate aplicațiile ulterioare trebuie rescrise pentru maşina virtuală.

La nivel sistem de operare, maşina virtuală rulează o mulțime de procese de domeniu utilizator. Aceste domenii de utilizator sunt separate. 'Kernel'-ul este acelaşi pentru toate domeniile de utilizatori.

La nivel sistem, există o serie de mecanisme de virtualizare avansată ('live migration', 'memory management', 'snapshots'). 'Live migration' poate implica procese, maşini virtuale, stocări. Mecanisme privind gestiunea memoriei pot implica compresii de date, mecanismul de transfer Delta, deduplicarea datelor, partajarea paginilor (4 KB pe x86). Mecanismul 'snapshot' se referă la starea unui sistem într-un anumit moment de timp şi implică salvarea întregii stări (memorie şi stocare) a unei maşini virtuale. Se permite revenirea la o stare anterioară (o modificare greşită a configurației implică revenirea la 'snapshot').

Prin virtualizare, aplicațiile rulează într-un mediu sigur şi, astfel, securitatea, va fi îmbunătățită. De asemenea, virtualizarea rezolvă problema dependenței de platformă a aplicațiilor.

4. Virtualizarea în cadrul Volunteer Computing

Cu toate că s-au obținut rezultate ştiințifice cu ajutorul numărului uriaş de voluntari, cele mai multe grupuri de cercetare au continuat să utilizeze centre de calcul mari. Oricum, problema este nu a numărului (mare) de voluntari ci a numărului (mic) de proiecte care utilizează această resursă. Printre motive ar fi faptul că puterea calculului voluntar este utilizat pentru scopuri simultane şi rezolvarea de sarcini care necesită unități centrale de calcul pentru calcul puternic mai degrabă decât pentru gestiunea şi prelucrarea datelor. De aceea, 'middleware'-ul BOINC este specializat pentru acest gen de sarcini. Alt motiv pentru care nu se utilizează resursele calculului voluntar este acela legat de efortul de portare a unui proiect în acest mediu. Costurile de intrare şi întreținere pentru utilizarea BOINC implică: adaptarea 'software'-ului la fiecare platformă voluntară (adică, portabilitatea) şi legarea acestuia cu bibliotecile BOINC (acesta poate fi un mediu foarte eterogen, pornind de la 'hardware' diferit până la sisteme de operare diferite) , crearea unui sistem de furnizare a 'job'-urilor, re-crearea 'software'-ului pentru proiecte de fiecare dată când se modifică codul experimentului, încapsularea în mici 'task'-uri care pot fi trimise şi executate în mod independent unul de altul, achiziția şi întreținerea unei infrastructuri care să găzduiască proiectul [8].

Astfel de factori sunt mult mai provocatori pentru anumite proiecte pentru care aplicațiile sunt foarte mari (~GB) şi au cicluri frecvente de actualizare (~zile), acestea fiind aplicații dificil de modificat.

Pentru proiecte care ridică astfel de probleme, CERN a inițiat un proiect pentru încapsularea

‘task’-urilor proiectului în interiorul unor maşini virtuale. Acest nivel de încapsulare rezolvă multe probleme legate de mediul eterogen al maşinilor utilizatorilor, de portarea proiectului pe infrastructura BOINC şi de manipularea transparentă a modificărilor dinamice ale ‘software’-ului. Datorită virtualizării, cercetătorii doar vor avea de considerat mediul furnizat de maşinile virtuale şi să porteze aplicația lor în acest mediu. Acest mecanism oferă o “interfață Cloud” pentru toate aplicațiile şi , de aceea, această tehnologie a fost numită Volunteer Cloud [8].

Principalele avantaje ale virtualizării sunt izolarea şi securitatea. Prin implementarea monitorului maşinii virtuale se creează un alt nivel izolator peste sistem. Utilizând acest nivel furnizorii pot avea o mai mare încredere în proiectele care implică calcul voluntar. Astfel, dacă apar erori de execuție, acestea nu pot afecta întreg sistemul.

Configurarea conform unor cerințe specifice/individuale apare ca un alt avantaj al virtualizării, maşinile virtuale aflându-se la un nivel înalt în acest sens. Se pot specifica parametri pentru maşina virtuală (referitor la unitatea centrală de calcul, memorie, sisteme de operare diferite – în acelaşi ‘server’ ‘hardware’pot co-exista mai multe sisteme de operare).

Altă caracteristică care este activată este suportul pentru moştenire de pe alte platforme care este un rezultat al specificațiilor individuale. Dacă este necesar de a pune în funcțiune un ‘software’ moştenit, acesta poate fi portat într-o maşină virtuală.

De asemenea, prin virtualizare se poate avea un control sporit asupra resurselor. Astfel, se pot defini parametri pentru maşina virtuală (precum capacitatea de stocare pe disc, dimensiunea memoriei cu acces aleatoriu). Prin definirea unor politici de planificare este posibilă limitarea utilizabilității resurselor. Se poate stabili şi un control dinamic asupra resurselor.

Avantajele utilizării virtualizării pot fi afectate dacă nu se poate oferi un nivel acceptabil de performanță. În mediile virtualizate, execuția aplicațiilor poate conduce la apariția fenomenului de ‘overhead’ care semnifică consumul în putere de calcul şi comunicații cu efect de degradare a performanței. Astfel, inițializarea imaginii maşinii virtuale, timpul de încărcare a maşinii virtuale, pornirea, pauza şi oprirea maşinii virtuale sunt aspecte funcționale care pot degrada performanța. Funcționalitatea virtualizării are implicații asupra performanței maşinii gazdă. Atunci când este combinată cu calculul voluntar pentru a realiza execuția ‘job’-urilor, funcționalitatea virtualizării devine o provocare deoarece Volunteer Computing urmăreşte să utilizeze resursele publicului într-un mod eficient pentru un efect maxim. Pentru a adopta tehnica virtualizării în calculul voluntar, trebuie stabilit impactul performanței virtualizării asupra ‘middleware’-ului calculului voluntar. În acest sens, au şi fost făcute estimări experimentale cu privire la fenomenul de saturație determinat de funcționalitățile virtualizării prin execuția unor ‘job’-uri în diverse scenarii [4].

5. Volunteer Cloud – rezultat al virtualizării

Sistemul Volunteer Cloud (Clouds@home) a fost considerat ca o nouă formă de calcul Cloud care îşi construieşte infrastructura din calculatoarele voluntarilor. Combinând calculul Cloud cu calculul Volunteer s-a construit o nouă paradigmă care poate include atât puncte de vedere comerciale cât şi voluntare. Ideea a constat în aplicarea tehnologiei virtualizării în cadrul resurselor calculului voluntar. S-au elaborat mecanisme care permit virtualizarea pentru aplicațiile care rulează pe platforma BOINC. Această nouă paradigmă a fost considerată ca un prim pas către realizarea Clouds@home [3].

Propunerea conceptului de Clouds@home prin reunirea Cloud Computing cu Volunteer Computing a fost definită în [7]. Conceptul are la bază ideea de construire a resurselor Cloud de calcul şi stocare din resursele oferite de voluntari. Caracteristicile Clouds@home cele mai importante sunt următoarele [3]: în comparație cu Cloud Computing, Cloud@Home se aplică la scală mai joasă (domeniul de contribuție poate fi de la un singur utilizator care îşi partajează calculatorul cu organizația de cercetare), securitatea (în acest tip de infrastructură Cloud, datele şi resursele sunt protejate de calculatoarele locale), fiabilitatea (deoarece Cloud@home este stabilit pe resurse care sunt mai mari decât cloud-ul astfel încât fiabilitatea sa poate fi comparată cu Grid Computing sau Volunteer Computing şi este mai mare decât în cazul cloud), interoperabilitatea

între cloud-uri (unul dintre cele mai importante scopuri ale cloud@home este interoperabilitatea între cloud-uri), rolul activ pentru utilizatori (utilizatorii cloud@home pot vinde sau cumpăra resurse şi, în comparaţie cu forma tradiţională de cloud, utilizatorii au un rol mai activ).

Prin Volunteer Cloud controlul resurselor nu mai este deţinut de companii comerciale ci de către utilizatori care pot lua decizii cu privire la care resurse şi câte anume sunt necesare pentru a fi utilizate în acest mod distribuit. Acest tip de calcul adresează problema interoperabilităţii care există în calculul Cloud şi, de asemenea, construieşte o infrastructură într-un mod uşor şi ieftin.

Pentru atingerea scopurilor sistemelor calculului Volunteer Cloud trebuie depăşite unele probleme [5]. O mare provocare, pentru deplasarea de la centrele de date Cloud scumpe către resursele voluntarilor, o constituie volatilitatea şi disponibilitatea resurselor. În calculul voluntar se poate întâmpla adesea ca sistemele de calcul să îşi înceteze brusc activitatea fie datorită deţinătorilor resurselor fie datorită unor probleme tehnice, precum căderea tensiunii de alimentare sau defectarea sistemului de calcul. Resursele sunt foarte eterogene, cu caracteristici tehnice foarte diferite (referitor la unităţi centrale de calcul, memorii, discuri, lăţimi de bandă ale reţelelor, rata de apariţie a defectelor). Aceste aspecte au efect direct şi major atât asupra performanţei şi cât în ceea ce priveşte planificarea. De aceea, toleranţa la erori a fost considerată ca un aspect important pentru realizarea calculului Volunteer Cloud. O altă problemă delicată este aceea legată de încrederea voluntarilor în încărcările de lucru/‘job’-urile lansate, existând posibilitatea apariţiei, drept consecinţă, de rezultate denaturate datorate interpunerii rău-intenţionate (ceea ce impune proceduri de verificare a corectitudinii rezultatelor). De asemenea, este important de a convinge voluntarii să îşi doneze resursele la un nivel diferit de accesibilitate decât în cadrul calculului voluntar deoarece resursele nu sunt utilizate doar pentru probleme ştiinţifice ci sunt utilizate şi de către furnizori comerciali.

6. Concluzii

Modelele de calculul paralel şi distribuit constituie baza acestor tehnologii. Calculul scalabil este diferit de calculul optimizat pentru performanţă. Cloud poate suporta încărcări de lucru Grid şi HPC dar nu este în sine un Grid în sens “clasic”. Infrastructurile de tip Cloud (precum cele ale Amazon.com şi Google) nu s-au realizat în spiritul concepţiei Grid. Mai mult, o zonă a Grid-ului, EPP impune cerinţe presante să fie acceptat de cloud-uri precum EC2. În plus faţă de suportarea încărcărilor de lucru de tip EPP care rulează pe cloud-uri obişnuite, unele cloud-uri pot, de asemenea, dezvolta o zonă proiectată specific pentru încărcari de lucru de tip HPC. Rezultă că tehnologia HSC a fost gândită şi proiectată ca să permită HPC.

Deci, deşi grid nu este cloud, există o inter-relaţionare. În plus, din punct de vedere comercial, a existat şi o mare oportunitate pentru furnizorii de cloud de a accepta acest segment de piaţă, un exemplu fiind Amazon care a investit sume foarte mari în acest sens. Cloud Computing este un calcul orientat-serviciu care oferă servicii prin intermediul virtualizării.

Volunteer Computing nu este Cloud Computing dar pentru depăşirea unor probleme generate de mediul eterogen al Volunteer Computing, s-au iniţiat proiecte de virtualizare pentru “ascunderea” resurselor fizice. Procesul de virtualizare a condus la rezolvarea unor probleme şi a creat un mediu de execuţie mai sigur, prin virtualizare fiind depăşită problema dependenţei de platformă a aplicaţiilor şi fiind sporit gradul de securitate.

Ideea de a aplica virtualizarea în cadrul Volunteer Computing, realizându-se infrastructuri ‘cloud-like’ pe baza resurselor voluntare, a condus la apariţia unei noi paradigme de calcul: Volunteer Cloud Computing. Această tehnologie se confruntă cu o serie de probleme majore (în principal, disponibilitatea resurselor de calcul şi convingerea voluntarilor de a oferi resurse la alt nivel de accesibilitate) deoarece are ca scop şi aspecte comerciale pe lângă cele legate de rezolvarea unor probleme ştiinţifice masiv computaţionale.

Trebuie remarcată evoluţia tehnicilor de virtualizare prin intermediul căreia s-au dezvoltat tehnologiile de calcul, devenind posibilă şi apariţia Volunteer Cloud. Sistemele de operare paravirtualizate care sunt instalate în fişiere în format disc standard şi care rulează fie nativ fie pe

hipervizoare compatibile şi interschimbabile care uzează de gestiunea asistată 'hardware' a unităţilor centrale de prelucrare, a memoriei şi a dispozitivelor de intrare/ieşire par a contura viitorul virtualizării.

BIBLIOGRAFIE

1. **ANDERSON, D.; NOV, O.; ARAZY, O.:** Volunteer Computing: A Model of the Factors Determining Contribution to Community-based Scientific Research. International World Wide Web Conference, 2010.
2. **ANDERSON, D. P.:** Volunteer Computing Vs Cloud Vs Grid Vs HPC. http://www.volunteer-computing.org/EN/.
3. **ANJOMSHOA, M.; SALLEH, M.:** Overview on Clouds@home: Virtualization Mechanism for Volunteer Computing. International Conference on Parallel and Distributed Computing Systems (PDCS), 2014.
4. **ANJOMSHOA, M.; SALLEH, M.; KERMANI, M. P.:** The Cost of Virtualization Implementation in Volunteer Computing. International Conference on Computer, Communications, and Control Technology (I4CT). 2014
5. **ANJOMSHOA, M.; SALLEH, M.; KERMANI, M. P.:** A Taxonomy and Survey of Distributed Computing Systems. Journal of Applied Sciences, 2015.
6. **BIAS, R.:** Grid, Cloud, HPC… What's the Diff? http://cloudscaling.com/blog/cloud-computing/. 2010
7. **DISTEFANO, S.; CUNSOLO, V. D.; PULIAFITO, A.:** A Taxonomic Specification of Cloud@Home. Advanced Intelligent Computing Theories and Applications, With Aspects of Artificial Intelligence, 2010.
8. **DORNEANU, D.:** Volunteer Computing at CERN – Sustainability Model Proposal. CERN Openlab, 2011.
9. **FOSTER, I.; KESSELMAN, C.; TUECKE, S.:** The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of High Performance Computing Applications, 15(3), 2001.
10. **HAFEEZ, M.; ASGHAR, S.; MALIK, U. A.; REHMAN, A.; RIAZ, N.:** Message Passing Framework for Globally Interconnected Clusters. Journal of Physics: Conference Series 331, 2011.
11. **MELL, P.; GRANCE, T.:** The NIST Definition of Cloud Computing. National Institute of Standards and Technology, Information technology laboratory, 2009.
12. **MUNTEAN, T.:** Parallel Operating System Architecture for Message Passing Parallel Computers. OSF'93, 1993.
13. **RITEAU, P.:** An Overview of Virtualization Technologies. University of Rennes 1, 2011.
14. **SHENG, D.; KONDO, D.; CIRNE, W.:** Characterization and Comparison of Cloud versus Grid Workloads. IEEE International Conference on Cluster Computing (CLUSTER), 2012.