CURS PRACTIC DE CALCUL GRID

M. Dulea[1], 2015

CUPRINS

- INTRODUCERE 1
- 1. DEFINITII GRID 2
- 2. BENEFICIILE UTILIZATORULUI GRID 4
- 3. SERVICII GRID 4
- 4. SECURITATE 4
- LAB1: Obtinerea unui certificat grid de utilizator 6
- LAB2: Inscrierea utilizatorilor intr-o organizatie virtuala 9
- 5. ARHITECTURA GRID 10
- 6. MIDDLEWARE GRID – DESCRIERE FUNCTIONALA 11
- 7. IMPLEMENTARI IN EMI-1 12
- LAB3: Utilizarea UI - trimiterea spre executie pe CE 15
- 8. MANAGEMENTUL FLUXURILOR DE LUCRU SI AL DATELOR 17
- LAB4: Trimiterea *job*-urilor spre executie prin WMS 20
- 9. FRAMEWORK-UL SOFTWARE DIRAC 22
- LAB5: Lansarea si executarea de job-uri prin intermediul DIRAC 22
- ANEXA 1: UTILITARE PENTRU TRANSFER SECURIZAT DE DATE 26
- ANEXA 2: CATEVA NOTIUNI SI COMENZI LINUX 28
- ANEXA 3: NOTIUNI DE UTILIZARE A EDITORULUI VIM 30
- ANEXA 4: JOB DESCRIPTION LANGUAGE (JDL) 30

INTRODUCERE

Modelarea si simularea computationala s-a impus in ultimele trei decenii ca metodologie stiintifica independenta, intermediara teoriei si experimentelor, cu numeroase aplicatii in cercetare si inginerie. A devenit indispensabila in numeroase domenii cum sunt proiectarea sistemelor care necesita studii de dinamica a fluidelor (CFD - *Computational Fluid Dynamics*), bioinformatica, caracterizarea materialelor noi (de exemplu nanostructuri), proiectarea medicamentelor (*drug design*), mecanica structurala, imagistica medicala, prelucrarea si analiza datelor din marile experimente ale fizicii energiilor inalte (*Large Hadron Collider* – LHC), etc.

In acest context s-au dezvoltat doua paradigme principale de calcul intensiv: calculul de inalta performanta (*High-Performance Computing* – HPC) si calculul cu rata mare de executie a job-urilor (*High Throughput Computing* – HTC).

Scopul HPC este executia eficienta a unor sarcini de calcul (*tasks*) intensive care sunt strans cuplate. Pentru realizarea comunicarii masive intre task-uri sunt necesare conexiuni cu latenta scazuta capabile sa partajeze rapid datele intre un numar mare de procesoare implicate in executia unui program. In acest scop se utilizeaza supercomputere si clustere de calcul paralel cu latenta scazuta, care sunt optimizate pentru maximizarea numarului de operatii pe secunda.

[1] Autorul ii multumeste d-lui dr. Ionut Vasile pentru contributia sa la realizarea cursului.

© Institutul National de C-D pentru Fizica si Inginerie Nucleara Horia Hulubei (IFIN-HH)

Utilizatorii HPC sunt interesati de viteza de executie a unui program, iar performanta se masoara in FLOPS (numar de operatii in virgula mobila pe secunda), task-uri/secunda, rate I/O.

Spre deosebire de HPC, HTC are drept obiectiv executia eficienta a unui numar mare de sarcini de calcul slab cuplate. Deoarece nu este importanta comunicarea intre task-uri, acestea pot fi executate pe resurse de calcul eterogene distribuite geografic. Sistemele HTC sunt optimizate pentru a maximiza capacitatea de rulare concurenta a cat mai multor job-uri (cuasi-) independente. Utilizatorii sunt interesati de numarul de job-uri executate cu succes intr-un interval relativ mare de timp (luna sau an).

Calculul grid este o varianta de HTC care a aparut la sfarsitul anilor '90 prin inglobarea unor tehnologii de calcul distribuit deja existente. Astazi este una dintre cele mai utilizate tehnologii de calcul in stiinta si inginerie, si a constituit baza pe care furnizorii de servicii de calcul au lansat tehnologia *cloud computing*.

Acest curs ofera o introducere in tehnologia grid si activitati practice pentru a invata cum se pot utiliza resursele si serviciile grid. Training-ul se desfasoara pe infrastructura de calcul a Gridului National pentru Fizica si Domenii Conexe (GriNFiC), care a fost dezvoltata si este administrata de catre Departamentul Fizica Computationala si Tehnologii Informationale (DFCTI) din IFIN-HH.

❖ 1. DEFINITII GRID

In perioada 1998-2002 s-au formulat mai multe definitii pentru grid, care au evidentiat diferite aspecte specifice privind ***infrastructura*** si ***serviciile*** oferite utilizatorilor.

◇ 1.1 Definitia initiala:

"*A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.*"

[1. Ian Foster, Carl Kesselman, 1998, *The Grid: Blueprint for a New Computing Infrastructure*]

Definitia se limiteaza la enumerarea unor caracteristici ale serviciului de acces la resurse:

Dependable = reliable (de incredere)

Consistent = solid

Pervasive = general

◇ 1.2 Definitiile ulterioare

... pun accent pe **partajarea coordonata a resurselor**: "***coordinated resource sharing*** *and problem solving in dynamic, multi-institutional* ***virtual organizations***."

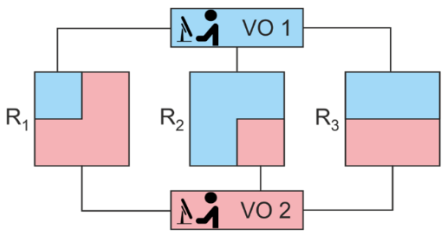
[2. Ian Foster, Steve Tuecke, 2000, *The Anatomy of the Grid*]

Resource sharing *= direct access to remote computers, software, data, and other resources + resource-brokering strategies* (brokering = ...)

> **Exemplu**: gridul poate oferi acces si control la distanta asupra unor dispozitive experimentale cu utilizatori multipli.

Resursele sunt in domenii administrative diferite si supuse diferitelor politici de acces si control.

Virtual organization *= dynamic collection of multiple organizations providing coordinated resource sharing* [3. Ian Foster, Carl Kesselman, Steve Tuecke, 2001, *The Anatomy of the Grid: enabling scalable virtual organizations, Lecture Notes in Computer Science, 2150, pp 200–222. Springer*]



◇ 1.3 Cerinte de baza pentru grid [4. Ian Foster, 2002, *What is the Grid? A Three Point Checklist*]

1) ***coordinates resources that are not subject to centralized control*** ...

(integrates and coordinates resources and users that live within different control domains—for example, the user's desktop vs. central computing; different administrative units of the same company; or different companies; and addresses the issues of security, policy, payment, membership, and so forth that arise in these settings. Otherwise, we are dealing with a local management system.)

2) … ***using standard, open, general-purpose protocols and interfaces*** …
(is built from multi-purpose protocols and interfaces that address such fundamental issues as authentication, authorization, resource discovery, and resource access. It is important that these protocols and interfaces be *standard* and *open*. Otherwise, we are dealing with an application specific system.)

3) … ***to deliver nontrivial qualities of service***.
(allows its constituent resources to be used in a coordinated fashion to deliver various qualities of service, relating for example to response time, throughput, availability, and security, and/or co-allocation of multiple resource types to meet complex user demands, so that the utility of the combined system is significantly greater than that of the sum of its parts.)

◇ 1.4 Definitii concise

Grid = un sistem de servicii care asigura, prin intermediul Internetului, accesul securizat al utilizatorilor la o infrastructura distribuita de calcul si stocare de date si partajarea dinamica a resurselor acesteia.

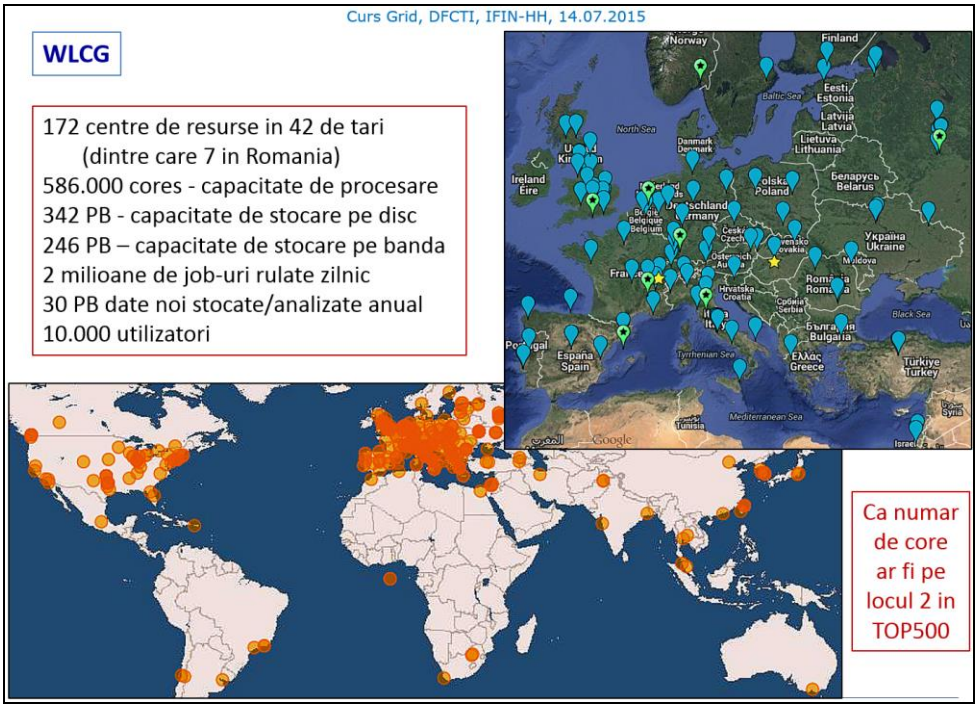
Exemple
WLCG (*Worldwide LHC Computing Grid*), http://wlcg-public.web.cern.ch/
OSG (*Open Science Grid*), http://www.opensciencegrid.org/
GriNFiC (*Gridul National pentru Cercetarea de Fizica si Domenii Conexe*), http://grid.ifin.ro

Infrastructura grid = un sistem distribuit geografic de resurse hardware si software neomogene care actioneaza coordonat pentru furnizarea serviciilor grid.

Exemplu: EGI (*European Grid Infrastructure*), http://www.egi.eu/

Organizatie virtuala (VO) = o comunitate dinamica de utilizatori sau institutii care beneficiaza de reguli identice de partajare a resurselor grid (de ex. pentru realizarea unor obiective comune).

Exemple (http://operations-portal.egi.eu/vo/search)
VO-urile suportate de WLCG pentru experimentele majore de la LHC-CERN: `alice`, `atlas`, `cms`, `lhcb`
VO-urile suportate de GriNFiC: `gridifin.ro`, `eli-np.eu`, `ronbio.ro`



❖ 2. BENEFICIILE UTILIZATORULUI GRID

- **Autentificare unica** (*single sign-on*) pentru acces la resurse multiple (pe timp limitat)
- **Accesarea transparenta a resurselor**. Utilizatorul nu are nevoie sa stie unde sunt resursele; grid-ul descopera automat resurse disponibile pentru rularea aplicatiilor si ii ofera sursele de date cele mai convenabile (de ex. ca viteza de accesare)
- **Resurse nelimitate**. Activitatea utilizatorului nu mai este limitata de capacitatile de calcul, prelucrare si/sau de stocare de date ale unui singur sistem
- **Acces la servicii software**. Nu mai este necesara instalarea de software de aplicatie pe masina proprie a utilizatorului, acesta putandu-le rula de pe un dispozitiv mobil
- **Fiabilitate**. Ansamblul sistemului continua sa furnizeze servicii chiar daca o parte din componente (resurse) au cedeaza.
- **Confidentialitate**. Transferurile de date sunt securizate prin criptare
- **Inlesnirea activitatii de colaborare**. Utilizatorii se pot conecta usor intr-o retea colaborativa fara sa necesite parole de acces (folosesc certificate digitale X.509).

❖ 3. SERVICII GRID

Sunt o extensie a serviciilor web. Serviciile web sunt in general utilizate pentru comunicarea in retea intre dispozitive hardware.

Serviciu web = aplicatie software accesibila *permanent* la o adresa de retea prin protocoale Internet, care foloseste XML pentru schimb de mesaje, descrierea tipurilor de servicii disponibile si descoperirea de servicii in WWW.

XML (*eXtended Markup Language*) = meta-limbaj de marcare, utilizat pentru transfer de informatii in retele de calculatoare (de ex. Internet).
Mesajele XML sunt transmise peste HTTP (deci fara sa fie blocate de firewall-uri) utilizand protocolul de comunicare **SOAP** (*Simple Object Access Protocol*, www.w3.org/2000/xp/Group/).

Exemple de servicii web se gasesc acolo unde este necesara comunicarea dintre un site web si o sursa de date pentru a actualiza informatia primita de utilizator (ex: rapoarte meteo).

Pentru a facilita rularea in grid-uri diferite a aplicatiilor dezvoltate intr-un anumit grid este necesara definirea unor standarde deschise (*open standards*).

Standardele grid sunt definite de *Open Grid Forum* (OGF, www.ogf.org, fost *Global Grid Forum*). OGF a dezvoltat **OGSA** (*Open Grid Services Architecture*), care descrie serviciile grid (de descoperire si management al resurselor, de securitate, etc.) ca servicii web ce trebuie sa satisfaca cerinte generale specifice mediului grid.

Detalii privind standardele in cap. 1 din [5. F. Magoules et al, 2009, *Introduction to Grid Computing*]

❖ 4. SECURITATE

Accesul la resursele grid se bazeaza pe increderea reciproca dintre furnizorii si utilizatorii acestora. Deoarece ideea de partajare a resurselor, specifica mediului grid, contrazice in principiu normele de securitate ale centrelor care furnizeaza resurse, se impune definirea clara de catre furnizori, de comun acord cu utilizatorii, a:

- Politicii de **acces** – cui i se permite sa utilizeze resursele/serviciile grid si in ce conditii;
- Procedurii de **autentificare** – pentru stabilirea identitatii utilizatorilor si a resurselor grid;
- Procedurii de **autorizare** – pentru impunerea regulilor de partajare a resurselor.

Utilizatorii grid-ului sunt interesati sa beneficieze de comunicare de date **sigura** si **confidentiala** in retele publice nesigure.

De asemenea, trebuie sa existe mecanisme de **delegare** a (unei parti a) privilegiilor utilizatorului catre alte entitati, pentru a permite unor procese sa comunice si sa actioneze in numele sau pe resurse gazduite in diferite centre/domenii de securitate.

O implementare a normelor de management al increderii propuse in etapele initiale de elaborare a OGSA este *Grid Security Infrastructure* (**GSI**, [Randy Butler et al, 2000, *A National-Scale Authentication*

Infrastructure]), care asigura servicii de autentificare, autorizare, de integritate a mesajelor si de delegare a acreditarii utilizand standarde deschise (*open standards*).

Functie	Cine utilizeaza	Prin ce se implementeaza
Acces la resurse/servicii	utilizatori, aplicatii software	politica de acces
Autentificare	utilizatori, resurse, servicii	certificate grid; semnatura digitala
Confidentialitate	utilizatori, servicii	criptarea datelor
Integritatea datelor	utilizatori, servicii	semnatura digitala
Autorizare	utilizatori, resurse, aplicatii	politica de partajare a resurselor
Delegarea acreditarii	servicii	certificate autentificate de utilizator

◇ 4.1 Autentificare, confidentialitate, integritate

GSI are la baza **criptarea cu chei publice** (conform ***Public Key Infrastructure*** - **PKI** [http://www.itu.int/rec/T-REC-X.509-201210-I/en, http://www.cgi.com/files/white-papers/cgi_whpr_35_pki_e.pdf], autentificand utilizatorii, serviciile si resursele prin intermediul **certificatelor digitale X.509** [https://www.rfc-editor.org/rfc/rfc2459.txt], si foloseste **protocolul SSL** (*Secure Socket Layer*, [http://www.rfc-base.org/txt/rfc-6101.txt]) pentru comunicarea de date in retele publice.

Concret, cheile sunt siruri de biti de lungime fixata generate de algoritmi de criptare cum este, de exemplu, algoritmul RSA (*Rivest, Shamir, Adleman*, [http://www.emc.com/emc-plus/rsa-labs/pkcs/files/h11300-wp-pkcs-1v2-2-rsa-cryptography-standard.pdf]). Ele sunt utilizate de algoritmii respectivi pentru criptarea/decriptarea datelor.

PKI presupune ca fiecare utilizator dispune de o pereche de chei: una publica si una privata (accesibila doar utilizatorului respectiv). Criptarea asimetrica asigura:

1) **Autentificare**: datele criptate de catre utilizator cu cheia sa privata pot fi decriptate doar cu cheia sa publica. Deci decriptarea reusita folosind cheia publica a utilizatorului confirma faptul ca datele respective au fost criptate cu cheia sa privata, care ii este accesibila doar acestuia; se considera ca astfel se confirma identitatea fizica a utilizatorului care a criptat datele.

2) **Confidentialitate**: datele adresate unei entitati se cripteaza cu cheia sa publica si pot fi decriptate doar de catre entitate cu cheia sa privata. Pentru transmiterea unui mesaj criptat de la *A* la *B*, *A* cripteaza mesajul folosind cheia publica a lui *B* si il trimite catre acesta. La primire, *B* decripteaza textul folosind cheia sa privata. Se considera ca astfel doar *A* si *B* pot citi continutul in clar al mesajului.

3) **Integritate**: verificarea integritatii mesajelor trimise in clar se face utilizand *functii hash* si *semnatura digitala*.

Functie hash (*hash function*) = algoritm care asociaza oricarui sir finit de biti un sir unic de biti de lungime fixa, numit *rezumat* (*hash*).

O functie hash este eficienta pentru criptare daca: 1) este dificil de inversat (*one-way function* - reconstituirea din rezumat a mesajului initial este practic imposibila); 2) este *rezistenta la coliziune* (sunt greu de gasit doua siruri de biti diferite cu acelasi rezumat); 3) se comporta ca o functie aleatorie (*random function* - rezumatul criptografic este extrem de sensibil la modificarea chiar si a unui singur bit din sirul de intrare).

Exemplul 1 (clasic): algoritmul MD5 (*Message Diggest v.5*, [https://www.ietf.org/rfc/rfc1321.txt]); genereaza rezumate cu lungimea de 128 biti (32 caractere hexazecimale). Aplicatie: criptarea parolelor in `/etc/shadow` din Linux. S-a demonstrat in 2004 ca MD5 nu este rezistent la coliziune.

Exemplul 2: *Secure Hash Algorithm* (SHA) este o familie de functii hash care s-a dezvoltat in mai multe versiuni (SHA-0, ..., SHA-3). Lungimea rezumatului variaza (de ex., SHA-2 poate furniza rezumate cu lungimea de 224, 256, 384 si 512 biti).

Semnatura digitala a unui text = este creata prin aplicarea unei functii hash asupra textului respectiv, urmata de criptarea rezumatului astfel obtinut folosind cheia privata a autorului textului.

Semnatura digitala probeaza atat integritatea unui mesaj cat si identitatea autorului acestuia. Pentru aceasta destinatarul *B* executa urmatoarele operatiuni:

1. receptioneaza de la autorul *A* textul mesajului in clar, semnatura digitala a mesajului (generata folosind cheia privata a lui *A*), specificarea functiei hash *H* si a algoritmului de criptare folosite de *A*;
2. aplica textului mesajului aceeasi functie hash *H*, obtinand rezumatul R_{Text};
3. decripteaza semnatura digitala primita folosind cheia publica a autorului *A*; rezulta rezumatul $R_{Semnatura}$ al textului (cel asupra caruia *A* a aplicat anterior functia hash *H* pentru a obtine semnatura digitala folosind cheia sa privata).
4. compara R_{Text} cu $R_{Semnatura}$; daca sunt identice rezulta ca textul nu a fost modificat.

PKI foloseste pentru autentificare certificate digitale X.509 eliberate de o **Autoritate de Certificare** (*Certification Authority* – **CA**) care este considerata de incredere in cadrul infrastructurii grid.

Certificat digital = un fisier text care dovedeste identitatea unei entitati ce poseda o cheie publica.

Entitatea poate fi o persoana fizica, o institutie, o resursa sau un serviciu. Toate entitatile care au incredere in CA-ul care a eliberat un certificat vor accepta certificatul respectiv ca dovada a identitatii detinatorului.

Un certificat X.509 este compus din doua parti. Prima parte include, printre altele, informatii privind algoritmului de semnare digitala a certificatului (de ex. `md5WithRSAEncryption`), datele de identificare a CA, perioada de valabilitate a certificatului (de ex. 1 an), datele de identificare ale detinatorului, cheia sa publica, algoritmul de generare a cheii sale publice (`rsaEncryption`). Cea de-a doua parte consta din semnatura digitala a primei parti de catre CA.

Pentru verificarea autenticitatii certificatului i se aplica primei parti a acestuia functia hash specificata in identificatorul semnaturii digitale (`md5`), si se compara rezultatul cu sirul de caractere care se obtine prin decriptarea semnaturii digitale folosind cheia publica a CA. Daca cele doua siruri de caractere sunt identice certificatul nu a fost falsificat.

Certificat proxy = certificat emis de o entitate pentru a delega catre o alta entitate, pe durata limitata, o parte din privilegiile pe care le are.

Proxy = imputernicire, procura. Proxy-ul consta dintr-o pereche noua de chei; cheia privata poate fi necriptata deoarece proxy-ul este valabil pe o durata relativ scurta de timp (de ex. 12 ore). Certificatul proxy nu este semnat de CA, ci de catre entitatea care l-a emis.

Proxy-ul este frecvent folosit pentru a da comenzi in grid. Prin folosirea lui se realizeaza autentificarea o singura data pentru o sesiune de lucru (***single sign-on***) si se evita expunerea certificatului de utilizator in medii mai putin sigure.

❖ LAB1: Obtinerea unui certificat grid de utilizator

CERINTE: cont pe server Linux; `openssl; putty; WinSCP; Anexa 2`

CA-ul infrastructurii grid nationale pentru cercetare si educatie este *RomanianGRID CA* [http://www.romaniangrid.ro/].

Identitatea fizica a solicitantilor de certificate dintr-o institutie/organizatie este verificata de CA prin intermediul manager-ului *Autoritatii de Inregistrare* (RA – *registration authority*) al institutiei, care este persoana autorizata de catre institutie si CA sa administreze toate certificatele grid emise pentru institutia respectiva.

Solicitantii de certificate grid trebuie sa cunoasca in prealabil politicile curente de certificare (CP = *Certificate Policy*) si de utilizare a certificatelor (CPS = *Certificate Practice Statement*) ale CA, care sunt publicate la adresa http://www.romaniangrid.ro/cpcps.htm.

In continuare se va descrie procedura pentru obtinerea unui certificat grid de utilizator folosind utilitarul `openssl` pe platforma Linux [http://www.openssl.org/docs/apps/openssl.html].

Structura instructiunilor `openssl` este urmatoarea:

`openssl` *`comanda`* `[`*`optiuni_comanda`*`]` `[`*`argumente_comanda`*`]`

unde *`comanda`* poate lua, printre altele, valorile:

`genrsa` - pentru generarea unei chei private RSA (punctul 4 de mai jos);

`req` – pentru generarea Cererii de Semnare a Certificatului (CSR);

`pkcs12` – pentru crearea fisierelor pkcs12, descrise mai jos.

Partea I

1. Mai intai, solicitantul verifica versiunea utilitarului `openssl` pe masina unde doreste sa genereze cheile, cu comanda

```
$ openssl version -a
```

Daca versiunea instalata este raportata ca fiind vulnerabila cu calificativul "*Severity: High*" in *OpenSSL Security Advisory* (care poate fi consultat la adresa https://www.openssl.org/news), se va face upgrade la versiunea cea mai recenta.

2. Solicitantul creeaza un director de lucru (de ex. `~/grid`, vezi **◇A2.1**), care va contine cheia privata, si restrictioneaza accesul la calculator si la directorul de lucru astfel incat cheia privata (`userkey.pem`) ce va fi generata sa fie protejata conform regulilor din CP/CPS.

3. Solicitantul genereaza cheia privata proprie cu lungime minima de 2048 biti.

De exemplu, pentru generarea cu `openssl` a cheii private pe 2048 biti folosind procedeul de criptare Rivest-Shamir-Adleman (RSA), solicitantul foloseste comanda:

```
$ openssl genrsa -out userkey.pem 2048
```

Rezulta fisierul `userkey.pem` (fisierele `.pem` sunt codificate in format base64), de forma:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA2/BUFvucLVsA7G+UlJYIo6wRx3wq4AZzK/flGAkpAl8nn3kk
.....................................................................
cKdu1ijGMTlOiZ0swy7OAD4/dziz3qZJXPzULFbtrxlp1dvOB+8vCA==
-----END RSA PRIVATE KEY-----
```

unde "..." reprezinta mai multe linii asemanatoare.

IMPORTANT: din motive de securitate, detinatorul va atribui cheii private (`userkey.pem`) drepturile de access 400 (`chmod 400 userkey.pem`) si va lua masuri pentru a nu fi cunoscuta de alte persoane.

4. Solicitantul genereaza in acelasi director *Cererea de Semnare a Certificatului* (CSR - *Certificate Signing Request*), care va fi ulterior adresata CA prin intermediul RA, cu comanda:

```
$ openssl req -new -key userkey.pem -out req.pem -subj
"/DC=RO/DC=RomanianGRID/O=IFIN-HH/CN=Prenume Nume"
```

Ultima parte a comenzii (*subiectul cererii* = DN = *Distinguished Name*) il identifica in mod unic pe solicitant, prin specificarea tarii, a CA, a organizatiei prin intermediul careia se face cererea, si a numelui (CN = *common name*).

OpenSSL va produce fisierul `req.pem`, de forma:

```
-----BEGIN CERTIFICATE REQUEST-----
MIICoDCCAYgCAQAwWzESMBAGCgmSJomT8ixkARkWAlJPMRwwGgYKCZImiZPyLGQB
.....................................................................
wiTxng==
-----END CERTIFICATE REQUEST-----
```

`req.pem` contine, pe langa informatia criptata privind DN, cheia publica care va fi inclusa in certificat.

5. Solicitantul copiaza CSR-ul (`req.pem`) pe un mediu de stocare electronic (memory stick sau echivalent), pe care il va inmana personal manager-ului RA.

Partea a II-a

6. Managerul RA transmite CA-ului CSR-ul printr-un e-mail semnat digital.

7. CA-ul verifica CSR-ul si, daca aproba cererea, emite un certificat X.509v3 (`numar.pem`, unde `numar` este un numar serial atribuit de CA), pe care il trimite printr-un e-mail semnat digital atat solicitantului cat si managerului RA. Pe langa cheia publica a solicitantului certificatul mai contine semnatura digitala a CA si, in clar, informatii privind identitatea detinatorului cheii publice, perioada de valabilitate a certificatului, algoritmii de criptare folositi, etc.

8. Detinatorul va incarca in directorul de lucru certificatul `numar.pem` si ii va atribui permisiunea de acces 444 (`chmod 444 numar.pem`). Astfel, detinatorul certificatului, grupul sau si oricare alt utilizator au doar drept de citire a certificatului, pentru a evita modificarea accidental sau intentionata a acestuia.

9. **Salvarea informatiei personale in format PKCS12**

Este util ca proprietarul certificatului sa-si arhiveze cheia privata si certificatul X.509 intr-un fisier unic, in formatul standard PKCS12 (*Performance Information Exchange Syntax*, https://tools.ietf.org/html/rfc7292). In acest scop va da comanda:

```
$ openssl pkcs12 -export -inkey userkey.pem -in numar.pem -name Prenume_Nume -out prenume_nume.p12
```

Este solicitata parolarea accesului la fisier:

```
Enter export password:
Verifying - Enter export password:
```

Proprietarul va proteja fisierul cu o parola puternica care nu va fi cunoscuta de alte persoane.

Spre deosebire de certificatul `numar.pem`, fisiereul `prenume_nume.12` este in intregime criptat.

10. **Generarea semnaturii digitale**

Folosind fisierul `prenume_nume.p12`, proprietarul poate genera semnatura digitala cu comanda:

```
$ openssl pkcs12 -export -in numar.pem -inkey userkey.pem -out signature.p12 -name "Prenume Nume"
```

Aceleasi masuri ca la pct. 9 se vor lua si pentru protejarea prin parola a semnaturii digitale.

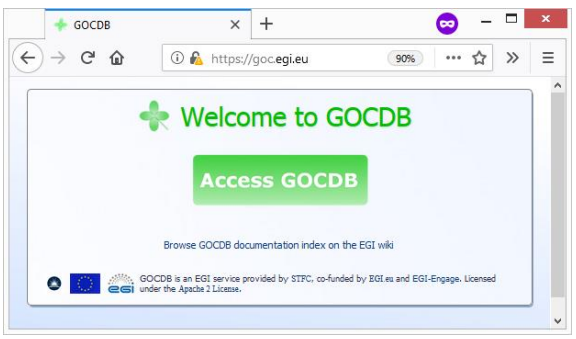
11. **Importul semnaturii digitale in browser**

Importul semnaturii digitale în browser este esential pentru autentificarea pe site-urile web care furnizeaza servicii ce deservesc infrastructura grid.

Procedura de import depinde de tipul browser-ului utilizat si de SO (Linux, Windows, Mac).

De exemplu, pentru Mozilla Firefox v. 38 pe platforma Windows se selecteaza *Options* > *Advanced* > *Certificates* > *View Certificates*, apoi se selecteaza butonul *Import* si se incarca fisierul `prenume_nume.p12`.

Test: https://goc.egi.eu



◇ 4.2 Autorizare

Utilizatorii autentificati pot beneficia de resurse si servicii grid doar daca sunt autorizati in acest scop.

Autorizare = verificarea privilegiilor unei entitati privind accesarea unei resurse sau a unui serviciu grid.

Autorizarea se face de catre administratorii resurselor/serviciilor respective, care permit accesul diferentiat la acestea pentru diferite organizatii virtuale (comunitati de utilizatori). Membrii fiecarei comunitati (VO) pot accesa doar resursele care au fost alocate VO-ului respectiv. In principiu, se admite ca un utilizator sa fie membrul mai multor VO-uri. In practica, este posibil ca in anumite cazuri acest lucru sa nu fie permis datorita politicii de comunitatilor stiintifice respective (cazul VO-urilor LHC).

La nivel global, administrarea informatiilor privind autentificarea si apartenenta utilizatorilor la diferite VO-uri se realizeaza prin intermediul serviciului **VOMS** (*Virtual Organization Membership Service*). Serviciul VOMS de autentificare si autorizare ruleaza pe servere VOMS dedicate (*VOMS servers*).

Serviciul VOMS foloseste atat o baza de date ce pastreaza informatia de autorizare a utilizatorilor (rolurile si capabilitatile lor), cat si instrumente software de accesare si operare a acesteia, prin intermediul carora se pot genera acreditari. Aceasta structura determina impartirea serviciului in doua componente majore:

- serviciul de administrare VOMS, folosit de managerul VO-urilor pentru administrarea acestora si a informatiei privind apartenenta utilizatorilor la VO-uri;
- serviciul de baza VOMS, prin intermediul caruia se emit acreditari pentru clientii autentificati.

Acreditarile emise contin atat informatia de autentificare necesara certificatelor proxy, cat si informatia specifica VOMS privind rolurile si capabilitatile utilizatorilor. Aceste date pot fi necesare aplicatiilor grid pentru autorizare sau in alte scopuri.

Serviciul VOMS care gestioneaza datele VO-urilor administrate de o organizatie/institutie este gazduit pe un server dedicat (*VOMS server*). REFERINTE: http://italiangrid.github.io/voms/ ; https://documents.egi.eu/public/RetrieveFile?docid=79&version=6&filename=EGI-SPG-VOManagement-V1_0.pdf
VOMS Attribute Certificate Format standard https://www.ogf.org/documents/GFD.182.pdf
https://wiki.italiangrid.it/twiki/bin/view/VOMS/VOMSSystemAdministratorGuide

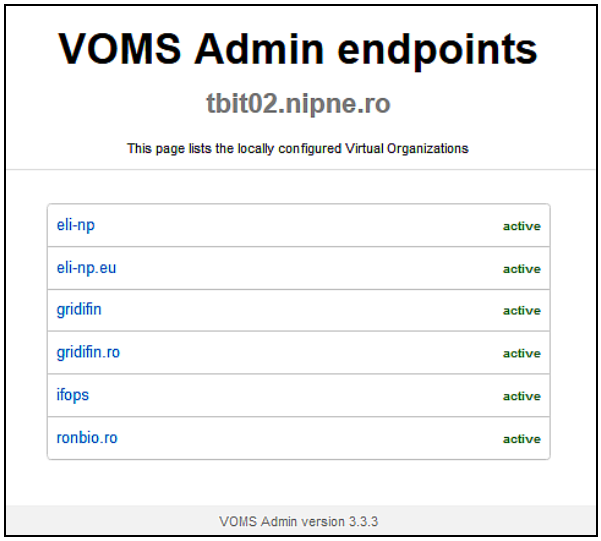
❖ LAB2: Inscrierea utilizatorilor intr-o organizatie virtuala

CERINTE: certificat grid emis de un CA acceptat de catre infrastructura grid

DFCTI suporta, prin intermediul site-ului grid GRIDIFIN, trei organizatii virtuale:

- **`eli-np.eu`**, http://operations-portal.egi.eu/vo/view/voname/eli-np.eu, VO global dedicat comunitatii de utilizatori din proiectul ELI-NP;
- **`gridifin.ro`**, http://operations-portal.egi.eu/vo/view/voname/ronbio.ro, VO national care sustine cercetarea multidisciplinara in fizica si domenii conexe;
- **`ronbio.ro`**, http://operations-portal.egi.eu/vo/view/voname/ronbio.ro, care deserveste comunitatea nationala a cercetatorilor din domeniul biologiei computationale.

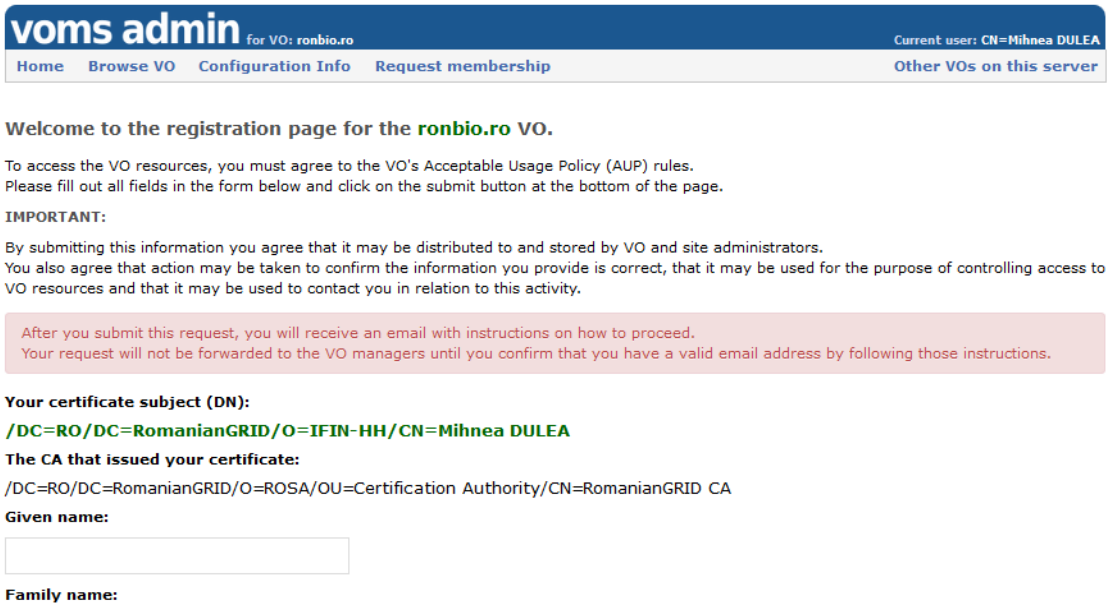
Cu browser-ul in care a fost importata semnatura digitala, utilizatorul care doreste sa se inscrie intr-o organizatie virtuala administrata de IFIN-HH acceseaza interfata web a serverului VOMS, la adresa https://tbit02.nipne.ro:8443/. Se deschide lista VO-urilor gestionate de IFIN-HH:



In continuare presupunem ca utilizatorul doreste sa se inregistreze in VO-ul **`ronbio.ro`**. In urma selectarii link-ului respectiv se afiseaza formularul de inregistrare in VO. Faptul ca utilizatorul a fost identificat de catre serviciul VOMS se confirma prin afisarea CN-ului din semnatura digitala, care apare in coltul din dreapta sus al paginii web.

Utilizatorul completeaza rubricile specificand numele, prenumele, institutia si datele de contact (inclusiv adresa de e-mail).

De asemenea, utilizatorul citeste si confirma acceptarea politicii de utilizare a VO-ului (*Acceptable Usage Policy* - AUP), dupa care trimite cererea de aderare catre serviciul de inregistrare in organizatia virtuala respectiva.



Imediat dupa trimiterea cererii utilizatorul primeste pe adresa de e-mail declarata o solicitare de confirmare a solicitarii sale, trimisa din partea serviciului de inregistrare ("**`VOMS Admin for VO ronbio.ro`**").

Mesajul de raspuns trimis de utilizator este analizat de administratorul VO-ului respectiv, care decide acceptarea sau respingerea solicitarii. Decizia este comunicata utilizatorului.

Daca cererea sa este acceptata, utilizatorul este autorizat sa foloseasca resursele alocate VO-ului.

❖ 5. ARHITECTURA GRID

Pentru descrierea formala a grid-ului s-a adoptat un model arhitectural in straturi (similar modelului OSI - *Open Systems Interconnection* conceput pentru retelele de date).

APLICATII	Aplicatii grid(ificate), API	Utilizatori/programatori
MIDDLEWARE	Software specific de implementare a grid-ului	Servicii grid
RESURSE	Capacitati de procesare, stocare, senzori, instrumente	Managementul local al centrelor de resurse
RETEA	Hardware de retea + protocoale de comunicare de date	Furnizori de servicii de retea

La nivelul retea (care asigura comunicarea de date intre centre de resurse prin tranzitarea WAN-urilor multiple) si la cel al resurselor hardware (care sunt in general distribuite geografic) nu exista coordonare intre diferitele entitati care participa la grid. Coordonarea diferitelor contributii locale se realizeaza la nivel de *middleware*.

Middleware-ul este in general un software care furnizeaza pentru aplicatiile utilizatorilor servicii suplimentare celor oferite de sistemul de operare; el este reprezentat ca intermediar intre sistemele de operare ale resurselor si aplicatii.

❖ 6. MIDDLEWARE GRID – DESCRIERE FUNCTIONALA

Functiile principale ale middleware-ului grid sunt de a asigura:

- **Securitatea** (autentificarea, autorizarea, administrarea acreditarilor, delegarea privilegiilor, *single sign-on*) [**Cap. 4**];
- **Monitorizarea si descoperirea informatiilor grid**
  Monitorizare = observarea modului de utilizare a resurselor si a serviciilor.
  Descoperire = gasirea resurselor corespunzatoare cu cerintele unui task.
  Functia de monitorizare si descoperire este asigurata de serviciile de informare (*information services*) care, la lansarea fiecarui job, cauta resurse si servicii disponibile pentru rularea acestuia. Serviciile utilizeaza o lista a resurselor grid si starea curenta a acestora.
  Pentru eficientizarea operatiunii de culegere a informatiilor se poate adopta o organizare ierarhica, astfel incat mai multe servicii de descoperire a informatiilor situate pe un nivel inferior furnizeaza date pentru un serviciu de pe nivelul imediat superior.
  Serviciile de informare pot fi chestionate direct de catre utilizatori pentru obtinerea listei resurselor disponibile sau pot publica starea curenta a resurselor prin intermediul unei interfete web, pentru a fi chestionata de catre planificatorul de job-uri (*job scheduler*).
- **Managementul resurselor**
  Planificatorul de job-uri selecteaza si aloca resursele care vor fi folosite pentru rularea fiecarui job, folosind informatia furnizata de catre serviciile de informare. Deciziile luate de planificator depind de mai multi factori cum sunt, de exemplu, nivelul de prioritate al job-ului respectiv sau dependenta acestuia de executia altor job-uri din sistem.
- **Managementul datelor**
  Cuprinde administrarea accesului securizat la date, a replicarii, migrarii si indexarii datelor, a managementul metadatelor (informatii despre date), etc. Face apel la algoritmi de planificare ale caror decizii tin cont de localizarea datelor (*data-aware scheduling*).

Serviciile oferite de middleware-ul grid pot fi impartite in servicii de baza (core) si servicii de nivel inalt/utilizator.

APLICATII		Aplicatii grid(ificate)	Utilizatori grid
MEDII PROGRAMARE		API, biblioteci, compilatoare	Programatori
MIDDLEWARE	Servicii nivel utilizator	Management resurse/date Planificarea job-urilor	Managementul resurselor globale
	Servicii de baza	Servicii de informare, executie job-uri la distanta, acces stocare, QoS, etc	Access la sistemul de resurse distribuite
	Infrastructura de securitate	Autentificare, autorizare, delegare privilegii, etc.	Securizare acces si comunicare
RESURSE		Capacitati de procesare, stocare, senzori, instrumente	Managementul local al centrelor
RETEA		Hardware de retea + protocoale de comunicare de date	Furnizori de servicii de retea

Alte servicii de baza (elementare):

- **Executie de job-uri** – asigura rularea pe resurse aflate la distanta a job-urilor trimise spre executie. Foloseste serviciile de securitate pentru utilizarea autorizata a resurselor grid.

- **Transfer de fisiere** – transmiterea fisierelor de date necesare pentru executia unui job catre nodul pe care executia are loc. Trebuie sa fie sigur, eficient si sa asigure transferul de fisiere mari in intervale rezonabile de timp pentru a nu periclita executarea job-ului respectiv (job-urile au limita de timp).

Alte servicii de nivel superior:

- **Management de job-uri** – coordonarea serviciilor de securitate, de informare si de transfer de fisiere, pentru executarea job-urilor complexe, care constau din multe task-uri si/sau manipuleaza cantitati mari de date. Asigura, printre altele, mecanisme de resubmitare automata a job-urilor
- **Management de fisiere** – controlul si monitorizarea de catre utilizatori a transferului de fisiere. Utilizeaza si servicii de replicare a fisierelor in diferite locatii, pentru a reduce transferul global de date in sistem.
- **Management informatii** – ofera atat informatii privind resursele globale grid si starea job-urilor, cat si informatii publicate de alte servicii. Este cuplat cu un serviciu de monitorizare.

❖ 7. IMPLEMENTARI IN EMI-1

European Middleware Initiative (**EMI**, http://www.eu-emi.eu/) este middleware-ul utilizat astazi, printre altele, in cadrul WLCG (si GriNFiC), care a fost creat prin contributiile a trei proiecte middleware,

- **ARC** (*Advanced Resource Connector*, http://www.nordugrid.org/middleware/),
- **gLite** (care a incetat sa mai fie suportat din 2013, http://grid-deployment.web.cern.ch/grid-deployment/glite-web/),
- **UNICORE** (UNiform Interface to COmputing Resources, http://www.unicore.eu/),

precum si a dezvoltatorilor software-ului de stocare **dCache** (https://www.dcache.org/).

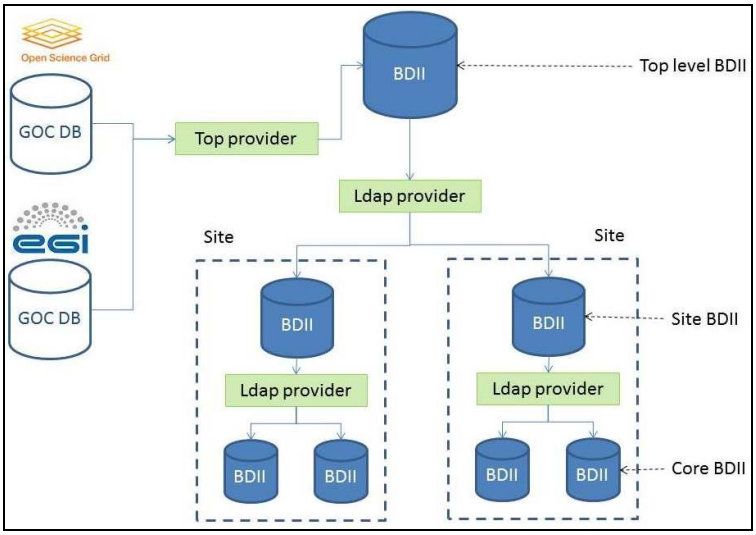
In continuare ne vom referi in special la versiunea gLite-EMI de implementare a serviciilor grid, care continua sa fie cea mai utilizata in EGI.

◇ 7.1 Serviciul de informare

Are o structura ierarhica pe trei niveluri care utilizeaza protocolul **LDAP** (***Lightweight Directory Access Protocol***) si sistemul de informatii ***Berkley Database Information Index*** (**BDII**).

LDAP este un protocol client-server conceput pentru accesarea, gasirea si actualizarea de date din directoarele de informatii, folosind protocolul internet (IP) si un limbaj specific pentru interogare sau modificarea datelor din director.

BDII este o versiune de server LDAP. Serverele LDAP interogheaza alte servere LDAP din ierarhie atunci cand nu dispun de informatia solicitata de catre client.



Cele trei niveluri ale serviciului de informare grid sunt:

- Nivelul resursa (*core*) – ofera informatii despre un serviciu grid care ruleaza pe resursa respectiva, furnizate chiar de catre serviciul respectiv;
- Nivel centru de resurse (*grid site*) – serverul site-BDII agregheaza informatiile de la toate core-BDII-urile care ruleaza in site-ul respectiv;
- Nivel top (*top level BDII*) – serverele top-BDII contin toate informatiile despre toate site-urile arondate, deci despre toate serviciile grid gazduite de acestea.

Ruland script-uri, furnizorii de informatii (*information providers - ip*) obtin informatii despre serviciile grid furnizate de resurse si le stocheaza in bazele de date OpenLDAP ale serverelor BDII ale resurselor. *ip*-urile *site-BDII*-urilor interogheaza *resource-BDII* si stocheaza rezultatele in bazele de date OpenLDAP ale *site-BDII*-urilor, etc.

Clientii sistemului de informare interogheaza un top-BDII pentru a gasi informatia necesara.

Detalii se gasesc la https://egee-uig.web.cern.ch/egee-uig/production_pages/QueryingInfoSystem.html

Informatiile generale privind site-urile grid din EGI sunt inregistrate in *Grid Configuration Database* (**GOCDB**, https://goc.egi.eu).

Atunci cand inregistreaza in GOCDB site-uri grid noi administratorii declara, printre altele, adresa URL a *site-BDII*-urilor acestora. *ip*-urile *top-BDII* dedicate site-urilor EGI descarca lista adreselor URL de pe GOCDB, dupa care aceste adrese a sunt utilizate pentru a interoga toate *site-BDII*-urile respective. Rezultatele interogarilor sunt folosite pentru popularea cu date a *top-BDII*.

Exemplu: O cautare a termenului "gridifin" pe site-ul web al GOCDB are ca raspuns pagina

Cu click pe numele site-ului se ajunge la pagina de informare privind GRIDIFIN, care include, printre altele, date de contact, locatia si tipurile de servicii grid oferite de site.

Mai sus este reprodusa lista tipurilor de servicii si a serverelor care le gazduiesc, din care aflam, de exemplu, ca serverul pe care ruleaza serviciul site-BDII este **`tbit06.nipne.ro`**.

Inainte de a descrie alte servicii, cum sunt WMS, UI, CREAM-CE si SRM, vom incerca sa aflam, cu utilitarul **`openldap`** dezvoltat de *OpenLDAP Software* (http://www.openldap.org/), ce resurse si servicii ofera site-ul GRIDIFIN.

Pentru aceasta, se chestioneaza, de exemplu, serviciul top-BDII de la CERN (furnizat de serverul **`lcg-bdii.cern.ch`**), cu comanda:

```
$ ldapsearch -x -h lcg-bdii.cern.ch -p 2170 -b Mds-Vo-name=GRIDIFIN,Mds-Vo-name=local,o=grid
```

Optiunile din comanda de mai sus au urmatoarea semnificatie: **`-x`** = autentificare simpla, **`-h`** = numele serverului BDII interogat (*host*), **`-p`** = port-ul de pe *host* pe care se face interogarea, **`-b`** = *base DN* = punctul de plecare de la care serverul incepe cautarea.

MDS = Monitoring and Discovery Service

information provider (IP) is defined as a service that interfaces a data collection service and provides information about the available resources to Grid Resource Information Service (GRIS).

The GRIS collects information about the resources from the IPs and makes the information available to the Grid Index Information Service (GIIS).

Raspunsul la interogarea din exemplu contine 2073 linii cuprinzand informatiile agregate furnizate de catre toate serverele core-BDII de pe site-ul GRIDIFIN.

◇ 7.2 Management local de job-uri si interfata cu utilizatorul

1. Compute Element (CE – Element de Calcul)

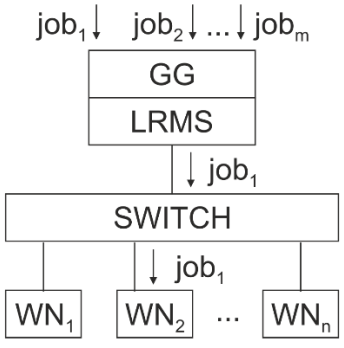
CE este un set de servicii furnizate de un server dintr-un site grid care asigura distribuirea job-urilor ce i-au fost repartizate catre cozile (*batch queues*) unui sistem local de management al resurselor (LRMS - *Local Resource Management System*).

CE este denumit si serverul care gazduieste serviciile CE; fiecare site grid dispune de cel putin un CE.

CE include: o interfata de acces a job-urilor catre resursele locale (*Grid Gate* – GG), un sistem local de management al resurselor (*Local Resource Management System* - LRMS) si resursele pe care se ruleaza efectiv job-urile.

Resursele pe care job-urile se executa pot fi chiar pe serverul CE dar, in majoritatea cazurilor, sunt gazduite pe serverele unui cluster de calcul numite noduri de lucru (*Worker Nodes* - WN).

WN-urile din cluster sunt conectate intr-o retea locala protejata printr-un firewall. Pe WN-uri sunt instalate componente middleware; software-ul de aplicatie specific VO-urilor suportate este instalat pe un sistem partajat de fisiere la care au acces toate WN-urile.



Distribuirea automata catre CE-uri a job-urilor submitate de utilizatorii grid este facuta de catre serviciul **WMS** (*Workload Management System*), care este descris mai jos in ◇ **7.3**.

Resursele pe care job-urile sunt executate se pot afla chiar pe CE dar, in marea majoritate a cazurilor, sunt gazduite pe nodurile de calcul (**worker nodes - WN**) ale site-ului/clusterului administrat de catre CE.

Practic, rolul principal al CE este de a repartiza job-urile primite spre executie pe WN-urile cu resurse disponibile care satisfac cerintele de rulare ale job-urilor. De asemenea, CE furnizeaza

informatii despre starea curenta a job-ului si, la terminarea executiei, despre resursele care au fost utilizate.

Doua implementari software actuale ale CE in EMI, utilizate in WLCG, sunt ARC-CE si **CREAM-CE** (*Computing Resource Execution And Management*). Cele doua implementari sunt in mare parte interoperabile; aici se va descrie CREAM-CE, care este deocamdata cel mai raspandit in WLCG.

CREAM-CE poate primi job-uri prin doua metode:

- trimise de utilizatori direct pe CE (folosind interfata CLI a CREAM-CE);
- trimise automat spre CE-ul respectiv prin intermediul *Workload Management System* (WMS) – care este descris mai jos.

LRMS (*batch system*) gestioneaza in cluster o coada de job-uri unica pentru fiecare CE. A fost implementat prin PBS (*Portable Batch System*), Torque (*Terascale Open-source Resource and QUEue Manager*) cu planificatorul Maui (http://www.adaptivecomputing.com), Slurm (*Simple Linux Utility for Resource Management*, http://slurm.schedmd.com/), etc.

Pe langa managementul solicitarilor de executie a job-urilor, CREAM-CE gestioneaza cererile de anulare a job-urilor, monitorizarea acestora, etc.

2. User Interface (UI)

UI este o suita de instrumente software si API-uri prin care se realizeaza accesul la servicii si programarea aplicatiilor grid.

UI este gazduit pe un server pe care utilizatorii au conturi ce contin certificatele lor personale.

Din contul de pe UI utilizatorul se poate autentifica si autoriza pentru a folosi resursele si serviciile grid si poate accesa sistemele de informare, de management de job-uri si de date.

UI dispune de o interfata in linie de comanda (CLI) prin intermediul careia se pot:

- lista resursele disponibile pentru executia job-urilor
- trimite job-uri spre executie
- anula job-uri
- afla starile job-urilor
- prelua datele de iesire ale job-urilor
- copia, replica si sterge fisiere din grid
- administra transferuri de fisiere
- afla starile resurselor, utilizand sistemul de informare

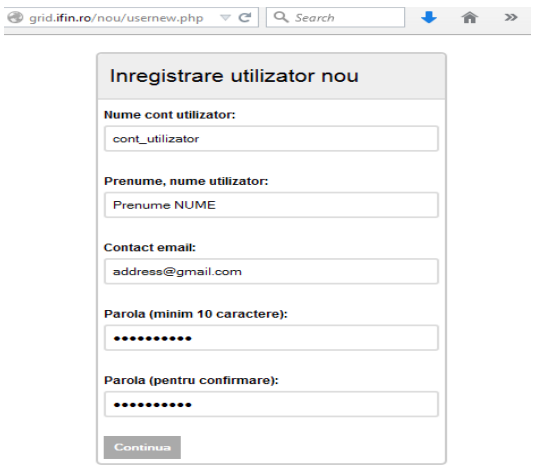
Conform paginii de informare privind GRIDIFIN (pag. 13), UI-ul site-ului GRIDIFIN este gazduit pe serverul **`tbitui.nipne.ro`**.

❖ LAB3: Utilizarea UI - trimiterea spre executie pe CE

CERINTE: **`usercert.pem`**, **`userkey.pem`**; notiunile JDL din ANEXA 4

1. **Crearea contului de utilizator pe UI**

Se poate solicita crearea de catre administratorul UI a unui cont pe **`tbitui.nipne.ro`** accesand interfata http://grid.ifin.ro/accesui.php



Se face click pe "*creare utilizator nou*".
Se completeaza datele care vor fi trimise administratorului sistemului **tbitui**.
Obs.: Pentru simplificarea activitatii, conturile cursantilor au fost deja create.

2. **Utilizarea UI - comenzi**

Dupa accesarea prin **ssh** a contului personal de pe **tbitui.nipne.ro**, utilizatorul creaza fisierul **~/.globus**, in care importa fisierele **usercert.pem** si **userkey.pem**; acestea se protejeaza la fel ca in LAB1.

Orice UI pune la dispozitia utilizatorului un set extins de instrumente software instalate.

Se poate vizualiza certificatul personal cu comanda:

```
$ grid-cert-info
```

Compatibilitatea dintre cheia privata si certificate se poate verifica cu comanda:

```
$ grid-proxy-init -verify
Your identity: /DC=RO/DC=RomanianGRID/O=IFIN-HH/CN=Prenume Nume
Creating proxy ........................................... Done
Proxy Verify OK
Your proxy is valid until: Thu Jul 23 06:57:40 2015
```

Autorizarea pentru lucrul in cadrul unui VO se face initiind un proxy, de exemplu:

```
$ voms-proxy-init -voms gridifin.ro
Contacting tbit02.nipne.ro:15005 [/DC=RO/DC=RomanianGRID/O=IFIN-
HH/CN=tbit02.nipne.ro] "gridifin.ro"...
Remote VOMS server contacted succesfully.
Created proxy in /tmp/x509up_u508.
Your proxy is valid until Thu Jul 23 07:02:18 EEST 2015
```

Informatii despre un proxy existent se pot afla cu:

```
$ voms-proxy-info –all
```

Comenzi pentru interogarea sistemului de informare

Aflarea tuturor CE-urilor care suporta un anumit VO (de ex. **gridifin.ro**), prin interogarea top-BDII:

```
$ lcg-info --vo gridifin.ro --list-ce
```

Raspuns:

```
- CE: tbit06.nipne.ro:8443/cream-pbs-gridifin.ro
```

Interogarea BDII privind situatia resurselor/serviciilor oferite de CE-uri pentru gridifin.ro

```
$ lcg-infosites --vo gridifin.ro ce
```

Raspuns:

```
# CPU Free Total Jobs  Running Waiting ComputingElement
----------------------------------------------------------------
  12   12     0           0       0     tbit06.nipne.ro:8443/cream-pbs-gridifin.ro
```

Informatiile listate cuprind numarul de CPU alocate, cate sunt libere, nr. total job-uri care ruleaza sau asteapta in coada, CE-ul.

3. **Utilizarea UI – submitarea *job*-urilor catre CE**

Exemplu: avem proxy creat si dorim sa submitam pe CE-ul GRIDIFIN job-ul prin **`test.jdl`**:

```
[
Executable = "test.sh";
StdOutput = "std.out";
StdError = "std.err";
InputSandbox = {"test.sh"};
OutputSandbox = {"std.out", "std.err"};
OutputSandboxBaseDestURI = "gsiftp://localhost";
]
```

Cele doua linii referitoare la **`OutputSandbox`** precizeaza ca fisierele de iesire trebuie sa fie copiate pe UI, iar fisierul **`test.sh`** contine urmatoarele instructiuni bash:

```
#!/bin/sh
clear
echo "Salut!"
echo "Rulez pe: "
/bin/hostname
/bin/sleep 20
echo "Pa!"
```

Comanda de submitare directa pe CE este:

```
$ glite-ce-job-submit -a -r tbit06.nipne.ro:8443/cream-pbs-gridifin.ro test.jdl
```

unde **`-a`** este necesar pentru a delega un proxy de utilizator, iar **`-r`** indica resursa (CE-ul) pe care se doreste submitarea.

Sistemul afiseaza id-ul job-ului si numele directorului creat pentru datele de iesire:

```
https://tbit06.nipne.ro:8443/CREAM358885629
```

unde se incarca fisierele **`std.out`** si **`std.err`**.

Continutul **`std.out`** :

```
$ cat std.out
Salut!
Rulez pe:
wngi3.nipne.ro
Pa!
```

Date despre starea job-ului se afla cu comanda

```
$ glite-ce-job-status https://tbit06.nipne.ro:8443/CREAM031904492
******  JobID=[https://tbit06.nipne.ro:8443/CREAM031904492]
        Status        = [DONE-OK]
        ExitCode      = [0]
```

❖ 8. MANAGEMENTUL FLUXURILOR DE LUCRU SI AL DATELOR

◇ 8.1 Workload Management System (WMS)

WMS este serviciul responsabil la nivel global de distribuirea si gestiunea sarcinilor de lucru pe resursele de calcul disponibile din grid.

WMS accepta task-uri trimise de utilizatori de pe UI, le aloca CE-urilor care sunt *cele mai adecvate*, tine evidenta starii job-urilor si recupereaza fisierele de output.

Alegerea celor mai adecvate CE-uri se realizeaza de catre un agent de distributie (*broker*) care consulta sistemul de informare privind indeplinirea simultana de catre CE-uri a urmatoarelor conditii:

- sa fie disponibile
- sa satisfaca cerintele exprimate de utilizator in fisierul .jdl
- sa poata accesa cel mai rapid fisierele de input specificate de utilizator.
- sa aiba rangul cel mai mare (rangul unui CE se calculeaza, de exemplu, in functie de numarul current al job-urilor care ruleaza si al celor care asteapta in coada)

Sarcinile de lucru procesate de WMS pot fi complexe, putand contine mai multe job-uri cu interdependente).

Serviciul de evidenta si contorizare (*Logging and Bookkeeping service* - LB) urmareste orice schimbare produsa in starea sarcinilor de lucru gestionate de WMS si inregistreaza starea si istoricul fiecarui job.

In gLite-EMI serviciul principal de management al job-urilor este **`WMProxy`**, care asigura:

- delegarea unui proxy pentru accelerarea operatiunilor
- brokering (gasirea CE-urilor adecvate job-urilor)
- submitarea task-urilor (job-uri unice sau multiple)
- anularea job-urilor
- recuperarea fisierelor de iesire

Trimiterea spre executie prin intermediul WMS a unui job descris de fisierul ***`job`*`.jdl`** se face cu comanda:

```
$ glite-wms-job-submit -a job.jdl
```

unde optiunea **`-a`** determina delegarea automata a unui proxy de utilizator catre serviciul **`WMProxy`**

◇ 8.2 Management de date

Datele in grid sunt continute fisiere care, din motive de siguranta si de optimizare a comunicarii, au de regula copii (replici) in mai multe centre de resurse.

Pentru a garanta ca toate copiile unui fisier inregistrat in grid au in permanenta un continut identic, se permite doar citirea sau stergerea acestora, nu modificarea lor.

Accesul utilizatorilor la date trebuie sa fie transparent (utilizatorii nu au nevoie sa stie unde sunt localizate datele); din acest motiv ei utilizeaza nume logice de fisiere (*logical file names* - **LFN**). Acelasi fisier poate primi din partea diferitilor utilizatori LFN-uri diferite.

Serviciile de management al datelor gestioneaza accesul la fisiere, inregistrarea unica a acestora si relatia dintre LFN-urile si locatia acestora.

Informatiile privind legaturile dintre identificatorii unici de inregistrare si LFN-uri sunt gestionate de serviciul Catalog de Fisiere, implementat in LCG prin *LCG File Catalogue* - **LFC**.

Stocarea sau descarcarea efectiva a fisierelor in/din centrele de resurse se face prin intermediul serviciului Element de Stocare (*Storage Element* - **SE**). Un fisier este considerat fisier grid doar daca este stocat pe un SE si inregistrat in LFC. Orice site grid poseda cel putin un SE.

Gestionarea traficului de fisiere intre SE-uri se realizeaza de catre Serviciul de Transfer de Fisiere (*File Transfer Service* - FTS)

◇ 8.3 Element de stocare

Un utilizator sau o aplicatie poate incarca, citi sau sterge fisiere de date (de) pe serverele de stocare din centrele grid doar prin intermediul serviciului SE. Pentru identificarea fisierelor, SE foloseste o baza de metadate (de exemplu cu sistem de management MySQL sau Oracle), care contine informatii despre acestea.

Fisierele de date pot fi doar citite; nu pot fi modificate decat prin stergere si inlocuire.

Exista mai multe tipuri de SE-uri, adaptate diferitelor cerinte si regimuri de lucru cu fisiere de date in grid.

GRIDIFIN utilizeaza *LCG Disk Pool Manager* (**DPM**), care permite adaugarea dinamica de discuri (HDD) pe servere eterogene conectate intr-o retea locala.

DPM mascheaza complexitatea arhitecturii sistemului fizic de stocare printr-un sistem de fisiere virtual. Astfel, transforma mai multe servere de disc (*data disk* (DD) - care asigura capacitatea de stocare) intr-o facilitate unitara de discuri – *Disk Pool*.

DPM foloseste un protocol specific pentru accesarea directa, securizata, la distanta (prin Internet) a fisierelor de pe SE-uri - (secure) RFIO (*Remote File Input/Output protocol*).

Alt exemplu de protocol pentru transfer de fisiere utilizat frecvent este *xroot*.

Managementul stocarii de date pe SE-uri se face prin intermediul protocolului SRM *Storage Resource Manager*.

SRM ii ofera utilizatorului o interfata prin care poate sa solicite fisiere, sa rezerve spatiu pe disc pentru adaugarea ulterioara de noi fisiere, sa pastreze temporar fisiere in buffer, etc., fara sa-i exhibe acestuia complexitatea sistemului fizic de stocare.

◇ 8.4 Infrastructura hardware. Exemple de site-uri

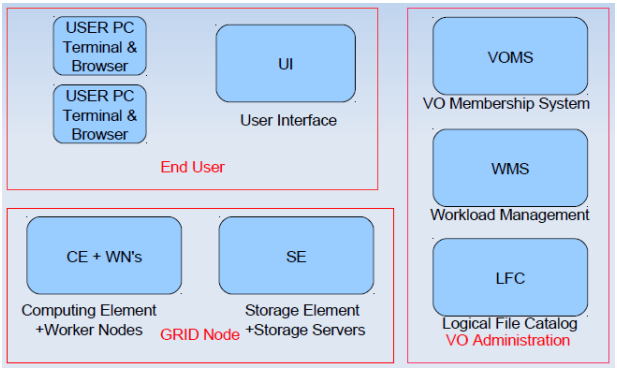
Sumarizand cele prezentate anterior, principalele componentele hardware ale infrastructurii grid cuprind: a) reteaua de date; b) elemente de calcul (*compute elements*) si nodurile de lucru (*worker nodes*) asociate acestora; c) elemente de stocare (*storage elements*) si unitatile de stocare asociate acestora; d) servere UI; e) serverele de administrare a serviciilor grid.

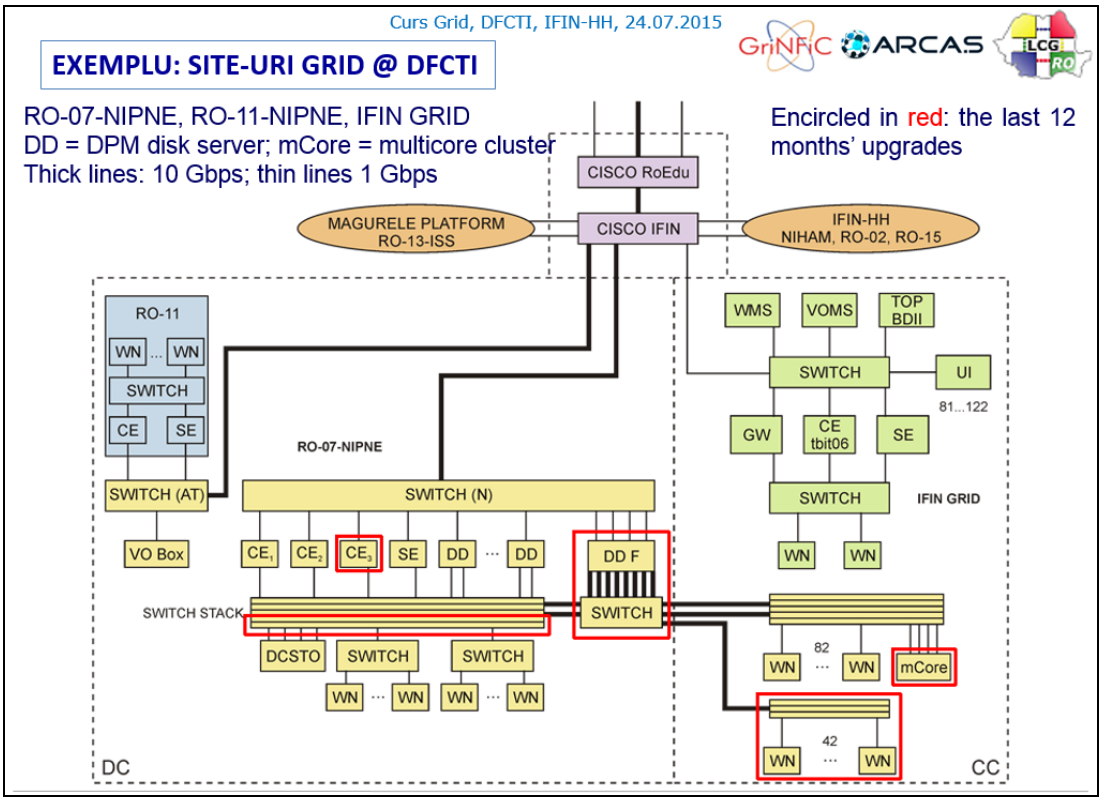
Intr-o reprezentare schematica a unitatilor hardware implicate direct in gestiunea job-urilor si a datelor in cadrul unei organizatii virtuale (VO), avem urmatoarele categorii:

a) hardware direct accesibil utilizatorului final (PC/laptop utilizator si serverul UI);

b) componentele hardware ale centrului grid (CE + WN-uri, SE + DD-uri);

c) componentele sistemului de administrare a VO-ului respeciv (VOMS, WMS, LFC)

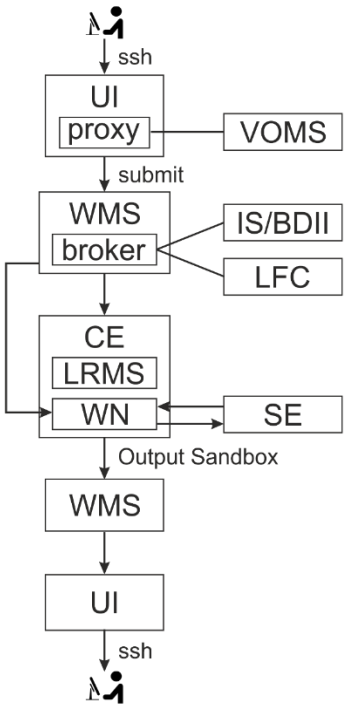




◇ 8.5 Fluxul de executie 'PUSH'

1. **Accesul utilizatorului pe UI** (prin ssh)
2. **Creare certificat proxy** (utilizeaza VOMS)
3. **Submitare job** -> WMS, cu preluare Input Sandbox
4. **WMS** cauta CE adecvate interogand IS/BDII si localizeaza fisiere input interogand LFC
5. **WMS** pregateste job-ul pt. submitare creand script *wrapper* care va fi trimis catre CE adecvat
6. **Planificare**: CE primeste job-ul si il trimite LRMS
7. **Executie job**: LRMS preia executia pe WN:
   Fisiere din Input Sandbox copiate pe WN
   Management date: copiere fisiere din SE pe WN
   Fisiere output -> SE si inregistrate in LFC
8. **Transfer rezultate**: fisiere OutputSandbox -> WMS
9. **Recuperare rezultate** de pe UI

WMS are posibilitatea sa resubmiteze job-ul catre alt CE daca CE-ul initial nu-l poate prelua.



❖ LAB4: Trimiterea *job*-urilor spre executie prin WMS

In cadrul LAB2 s-au inregistrat utilizatori in VO-urile **`gridifin.ro`** si **`ronbio.ro`**.

In partea a 3-a a LAB3 s-a trimis spre executie pe CE-ul GRIDIFIN un job, utilizand **`test.jdl`**.

1. CREARE PROXY

In LAB4 utilizatorii inregistrati in cele doua VO-uri vor trimite un job spre executie prin WMS. Pentru aceasta, creeaza mai intai proxy-uri cu:

```
$ voms-proxy-init -voms gridifin.ro
```

respectiv

```
$ voms-proxy-init -voms ronbio.ro
```

2. EDITARE .JDL CU SPECIFICAREA POSIBILITATII DE RETRIMITERE IN EXECUTIE

Se doreste submitarea prin WMS a job-ului descris prin **`test1.jdl`**, care se obtine din fisierul **`test.jdl`** prin copiere (**`$ cp test.jdl test1.jdl`**), urmata de editarea cu **`vim`**, comentarea ultimei linii si adaugarea altor doua:

```
[
Executable = "test.sh";
StdOutput = "std.out";
StdError = "std.err";
InputSandbox = {"test.sh"};
OutputSandbox = {"std.out", "std.err"};
#OutputSandboxBaseDestURI = "gsiftp://localhost";
RetryCount = 0;
ShallowRetryCount = 3
]
```

In cazul esecului executiei unui job in grid (datorita grid-ului, nu a programarii eronate), WMS poate retrimite automat job-ul respective in executie.

Exista doua moduri de retrimitere in executie a job-urilor de catre WMS:

- deep resubmission = atunci cand job-ul esueaza dupa ce incepe executia pe WN
- shallow resubmission = atunci cand job-ul esueaza inainte de a ajunge pe WN

`RetryCount` = numarul maxim permis de deep resubmissions

`ShallowRetryCount` = numarul maxim permis de shallow resubmissions

Se recomanda dezactivarea *deep resubmission*, cu **`RetryCount = 0;`**

pentru a evita situatiile:

a) WMS poate considera in mod eronat ca un job inca activ a esuat;
b) un job esueaza pe WN dupa ce a produs deja rezultate partiale (de ex. fisiere grid)

In ambele cazuri submitarea din nou a aceluiasi job poate produce inconsistente.

3. TRIMITEREA IN EXECUTIE PRIN WMS

Comanda de trimitere spre executie a job-ului **`test1`** prin WMS este:

```
$ glite-wms-job-submit -a test.jdl
Connecting to the service
https://tbit04.nipne.ro:7443/glite_wms_wmproxy_server
================ glite-wms-job-submit Success ======================
The job has been successfully submitted to the WMProxy
Your job identifier is:
https://tbit04.nipne.ro:9000/P0Ze6MCchfQ482igjpBdhA
====================================================================
```

4. MONITORIZAREA STARII JOB-ULUI

Pentru a afla starea job-ului, se tasteaza comanda:

```
$ glite-wms-job-status job_ID
================== glite-wms-job-status Success ====================
Status info for the Job : https://tbit04.nipne.ro:9000/shu4feNJSc1AJSyKIQ8NkQ
Current Status:     Running
Status Reason:      unavailable
```

Odata cu finalizarea job-ului, starea acestuia se va modifica in:

```
Current Status:     Done(Success)
Logged Reason(s):
    - job completed
    - Job Terminated Successfully
Exit code:          0
Status Reason:      Job Terminated Successfully
```

5. RECUPERAREA FISIERELOR CU REZULTATE

Preluarea fisierelor de output se realizeaza cu comanda:

```
$ glite-wms-job-output job_ID
```

Spre deosebire de cazul submitarii directionate manual catre un anumit CE, output-ul este transmis in acest caz in directorul **`/tmp`**:

```
have been successfully retrieved and stored in the directory:
/tmp/jobOutput/utilizator_job_ID
```

Cea mai simpla modalitate de a citi rezultatul este prin comanda:

```
cat /tmp/jobOutput/utilizator_job_ID/std.out
```

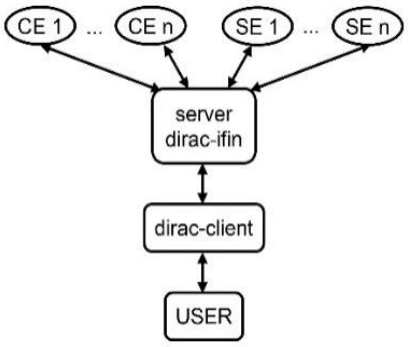
❖ 9. FRAMEWORK-UL SOFTWARE DIRAC

Framework-ul software **DIRAC** (*Distributed Infrastructure with Remote Agent Control*, [http://diracgrid.org/]) reprezinta o solutie de management unitar a activitatii utilizatorilor locali, care le ofera acestora o interfata web pentru rularea job-urilor pe diferite VO-uri si pe sisteme de resurse eterogene cum sunt site-uri grid, clusteri HPC sau sisteme cloud.

Prin intermediul DIRAC se poate face monitorizarea resurselor si administrarea job-urilor intr-un mod mai simplu decat printr-un UI (*user interface*) local.

DIRAC dispune de un WMS propriu, de sisteme de management pentru date, pentru cererile utilizatorilor si pentru stocare, de instrumente de monitorizare si contorizare a activitatii, putand sa realizeze statistici privind job-urile finalizate cu succes sau cele care au esuat.

De asemenea, prin DIRAC utilizatorii pot efectua preprocesari ale job-urilor pe resurse locale, inainte de a le trimite in griduri de productie (cum este WLCG).



DIRAC WMS este compus din servicii centrale si agenti care opereaza la distanta (remote agents - sunt aproape de resurse si intermediaza interactiunea lor cu serviciile centrale).
Mecanismul de executie este "Job Pulling" (https://github.com/DIRACGrid/DIRAC/wiki/JobManagement).

◇ 9.1 Fluxul de executie 'PULL'

WMS dispune de o coada centrala de task-uri (*Central Task Queue*), **singura din sistem**.

Job-urile din coada declanseaza trimiterea de agenti pilot (*Pilot Agents*) in grid.

Agentii pilot ruleaza pe WN-uri cu sarcinile:

- rezerva resurse pt. executia imediata de job-uri

- solicita de la WMS job-uri pt. executie

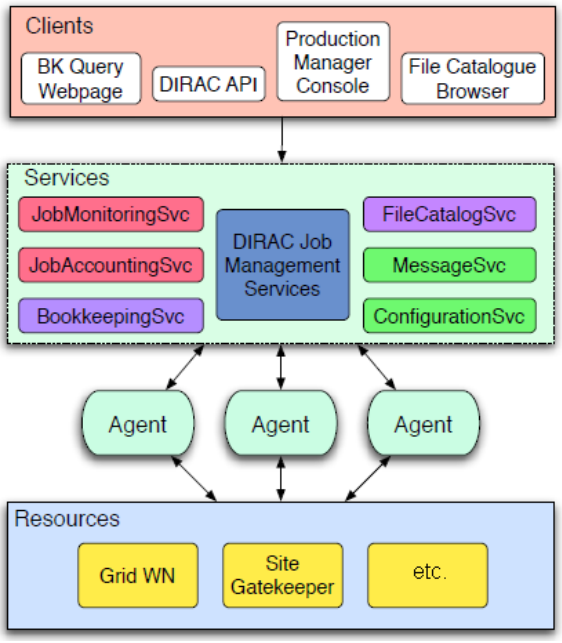
- management executie si transfer de date catre SE

Agentii pilot 'trag' (***pull***) job-uri din WMS catre WN

La sosirea pe WN agentul pilot verifica mediul local.

Daca resursele hard/soft nu corespund, raporteaza problema catre WMS, apoi moare si elibereaza resursele.

Deci job-urile propriu-zise nu sunt compromise de eventualele probleme de pe WN-uri.



❖ LAB5: Lansarea si executarea de job-uri prin intermediul DIRAC

CERINTE: `ssh`, `scp`, `vim`, cont pe serverul `dirac-client.nipne.ro`, certificat grid si cheia privata.

Modalitatea optima de acces al utilizatorilor la software-ul DIRAC este prin intermediul serverului **`dirac-client`**. In acest scop administratorul serverului creeaza cate un cont pentru fiecare utilizator si le comunica acestora datele de acces. In continuare prin "*directorul radacina al utilizatorului*" se va intelege directorul alocat acestuia (**`/home/nume_cont`**, pe scurt **`~`**)[Anexa 2.1].

Utilizatorii vor accesa conturile prin intermediul `ssh` (`putty`) si isi vor schimba parolele initiale cu alte parole cu acelasi grad de dificultate, folosind comanda **`passwd`**.

Procedura de instalare si configurare a clientului DIRAC de catre fiecare utilizator este urmatoarea:

1. Utilizatorul isi creeaza in directorul radacina subdirectorul `.globus`, cu comanda `mkdir ~/.globus`

2. Isi copiaza certificatul grid si cheia privata (in format **.pem**) in directorul **.globus**, folosind **scp** (**WinSCP**). Schimba, daca este necesar, numele certificatului in **usercert.pem** si numele cheii private in **userkey.pem**. Aceste fisiere vor fi necesare mai tarziu pentru a crea proxy-uri.

3. Seteaza obligatoriu permisiunile de acces la `444` pentru certificatul grid si la `400` pentru cheia privata.

4. Creeaza in directorul radacina un director (de ex. **dirac**) in care va instala software-ul de client

2. Descarca in directorul respectiv pachetul de instalare a clientului DIRAC, folosind utilitarul **wget**, cu comanda:

```
$ wget -np -O dirac-install
https://github.com/DIRACGrid/DIRAC/raw/integration/Core/scripts/dirac-install.py
--no-check-certificate
```

(downloader-ul **wget** preia de pe site-ul github.com pachetul fara a fi necesara verificarea certificatelor; acestea se verifica in momentul in care se dau comenzile specifice DIRAC).

3. Fisierul **dirac-install** descarcat devine executabil cu comanda

```
$ chmod +x dirac-install
```

4. Utilizatorul executa fisierul, prin comanda

```
$ ./dirac-install -r versiune
```

(unde ***versiune*** reprezinta versiuniea curenta a serverului **dirac-ifin**, vizibila in coltul din stanga jos al portalului web https://dirac-ifin.nipne.ro/DIRAC/ (de ex. **v6r13-pre34**). In urma instalarii directorul ***dirac*** este populat cu subdirectoare si fisiere, dintre care **bashrc** contine instructiuni pentru setarea mediului de lucru.

5. Prin comanda **source bashrc** se seteaza mediul de lucru pentru a putea fi utilizate comenzile DIRAC.

6. Inainte de a configura clientul DIRAC este necesar ca utilizatorul sa creeze un proxy, prin comanda **dirac-proxy-init -x** (optiunea **-x** se foloseste doar pentru configurare, nu si pentru utilizarea clientului DIRAC).

7. Configurarea clientului DIRAC se face cu comanda:

```
$ dirac-configure -V VO -S setup -C dips://dirac-
ifin.nipne.ro:9135/Configuration/Server
```

unde ***VO*** reprezinta organizatia virtuala de care apartine utilizatorul (de ex. **gridifin.ro**), iar ***setup*** - setup-ul pe care se va lucra (de ex. **GRIDIFIN-DIRAC-Production**, care este vizibil in coltul din dreapta sus al portalului web).

9. Utilizatorul creeaza un proxy, care va permite lansarea comenzilor specifice DIRAC, prin comanda **dirac-proxy-init**. Din acest moment poate incepe trimiterea in executie a job-urilor.

Obs.: Daca in urma comenzii de mai sus se primeste mesajul

```
Generating proxy...
Can't contact DIRAC CS: Reason(s):
        Unauthorized query (User possibly not registered with dirac server)
```

se solicita interventia administratorului serverului DIRAC.

Daca totul decurge normal, apar afisate urmatoarele mesaje:

```
Generating proxy...
```

```
Failed to add VOMS extension: no standard grid interface available
Uploading proxy for dirac_user...
Proxy generated:
subject      : /DC=RO/DC=RomanianGRID/O=IFIN-HH/CN=Mihnea DULEA/CN=proxy
issuer       : /DC=RO/DC=RomanianGRID/O=IFIN-HH/CN=Mihnea DULEA
identity     : /DC=RO/DC=RomanianGRID/O=IFIN-HH/CN=Mihnea DULEA
timeleft     : 23:59:59
DIRAC group  : dirac_user
path         : /tmp/x509up_u502
username     : mdulea
properties   : NormalUser

Proxies uploaded:
 DN                                                | Group      | Until (GMT)
 /DC=RO/DC=RomanianGRID/O=IFIN-HH/CN=Mihnea DULEA | dirac_user | 2016/06/10 12:35
```

Pentru organizarea activitatii se recomanda urmatoarea structura de directoare: in directorul radacina utilizatorul creeaza directorul **jobs**, apoi in **jobs** va creea cate un folder pentru fiecare job (de ex., primul **.jdl** descris mai jos poate fi gazduit in **/home/user/jobs/job1**). Fisierele **.jdl** (de descriere a joburilor), se pot edita cu **vim** sau alt editor unix.

Exemplul 1: Fisierul **test1.jdl** pentru un job simplu, fara fisier de input:

```
[
JobName = "Simple_Job";
Executable = "/bin/ls";
Arguments = "-ltr";
StdOutput = "StdOut";
StdError = "StdErr";
OutputSandbox = {"StdOut","StdErr"};
]
```

Exemplul 2: Un job cu fisiere de input si output, descris in **test2.jdl**:

```
[
JobName = "IO_Job”;
Executable    = "test2.sh";
StdOutput     = "test2.out";
StdError      = "test2.err";
InputSandbox  = {"test2.sh"};
RetryCount = 1;
ShallowRetryCount = 1;
requirements=other.GlueCEInfoHostName=="tbit06.nipne.ro";
OutputSandbox = {"test2.out", "test2.err"};
]
```

Se creeaza fisierul de input, numit in acest caz **test2.sh**, care va contine:

```
#!/bin/sh
clear
echo "Salut! Sunt un job cu fisiere de intrare si iesire!"
echo "Rulez pe: "
/bin/hostname
echo "care are: "
/bin/uname -a
/bin/date
cat /etc/issue
echo "Rulez sub numele de utilizator: "
/usr/bin/whoami
/bin/sleep 20
echo "La revedere!"
```

Obs.: Daca se utilizeaza un editor dos, se va converti fisierul **.sh** editat in dos in formatul unix, folosind **dos2unix.**

Pentru rularea unui job se lanseaza din folderul corespunzator acestuia comanda:

$ dirac-wms-job-submit *nume_job*.jdl

In urma comenzii anterioare, utilizatorul va primi un *ID* pentru jobul trimis in executie. De exemplu:

JobID = 52

Pentru a afla starea job-ului identificat cu *ID*, se foloseste comanda

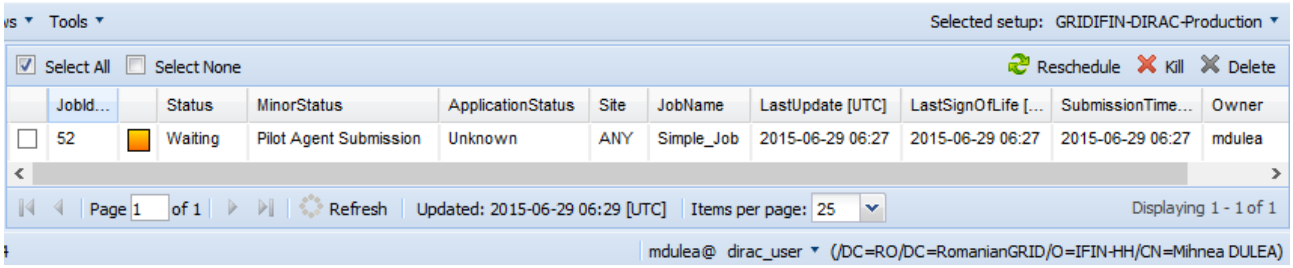
$ dirac-wms-job-status *ID*

In exemplul de mai sus:

$ dirac-wms-job-status 52

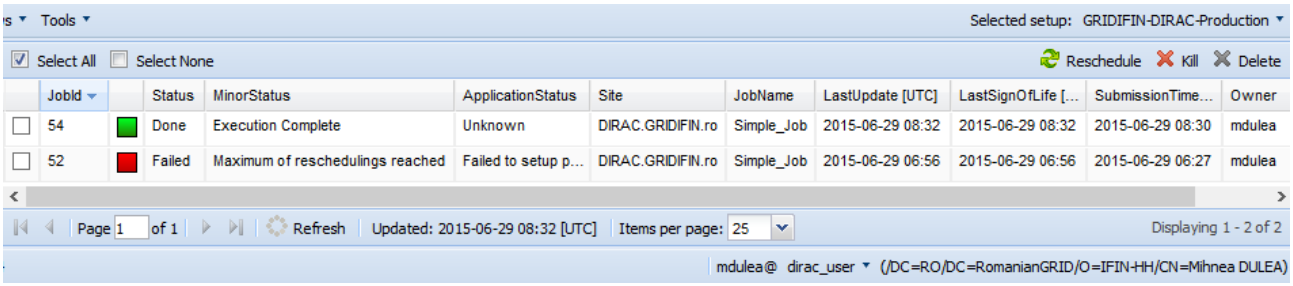
JobID=52 Status=Waiting; MinorStatus=Pilot Agent Submission; Site=ANY;

Starea job-ului poate fi monitorizata prin intermediul portalului. In acest scop utilizatorul se autentifica mai intai cu click pe link-ul din coltul din dreapta jos. Dupa autentificare isi poate afisa starea job-urilor selectand "*Job monitor*" din meniul "*Jobs*". Apare un sumar ca mai jos:



Daca trimiterea spre executie esueaza, sistemul repeta automat operatiunea de un numar de ori (care este fixat de administrator), dupa care renunta si afiseaza un mesaj de eroare. In exemplul de mai sus, deoarece utilizatorul s-a inregistrat intr-un VO necorespunzator, se afiseaza mesajul "**Failed to setup proxy: Error retrieving proxy**".

Dupa corectarea erorii, retrimiterea spre executie (sistemul atribuind de aceasta data **JobID=54**) si finalizare lista job-urilor se modifica astfel:



Pentru joburile care au fisier de output (Exemplul 2), dupa executie se poate vizualiza **path**-ul fisierului de output prin comanda

$ dirac-wms-job-get-output *ID*

~/jobs/job2/58/test2.out

Continutul fisierului se poate citi cu ajutorul comenzii

$ cat *path/nume_fisier*

In cazul Exemplului 2, continutul fisierului **test2.out** este urmatorul:

```
Salut! Sunt un job cu fisiere de intrare si iesire!
Rulez pe:
wngi3.nipne.ro
care are:
```

```
Linux wngi3.nipne.ro 2.6.32-504.23.4.el6.x86_64 #1 SMP Tue Jun 9 11:55:03
CDT 2015 x86_64 x86_64 x86_64 GNU/Linux
Mon Jun 29 13:55:43 EEST 2015
Scientific Linux release 6.6 (Carbon)
Kernel \r on an \m

Rulez sub numele de utilizator:
gridifin33
La revedere!
```

❖ ANEXA 1: UTILITARE PENTRU TRANSFER SECURIZAT DE DATE

Ati primit un nume de utilizator si o parola de acces la noul dvs. cont (*nume_cont*) de pe serverul *nume_server* (de ex. `dirac-client.nipne.ro`), cu sistem de operare (SO) Linux. Doriti sa accesati contul in mod securizat de pe PC-ul propriu. Pentru aceasta puteti folosi un protocol de retea criptat, cum este *Secure Shell* (SSH).

◇ SSH (http://www.snailbook.com/protocols.html)

Secure Shell este un protocol de retea care realizeaza transferul securizat de date intre un client si un server folosind criptografia cu chei asimetrice [**◇4.1**] pentru autentificarea serverului si, daca este necesar, a utilizatorului de catre server. Principala utilitate este accesul si executarea de comenzi pe server printr-o conexiune sigura. Pentru aceasta utilizatorul foloseste un client ssh.

◇ OpenSSH (http://www.openssh.com/)

Este o suita de programe ssh disponibila in majoritatea distributiilor Linux, care include clientul `ssh`, serverul `sshd` (care ruleaza ca *daemon* asteptand conectarea clientilor pe portul 22) si utilitarul `scp` (*Secure Copy*, descris mai jos).

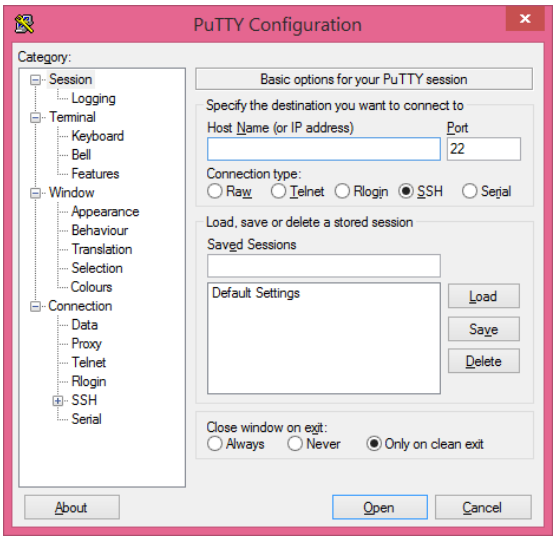
Pentru autentificare, utilizatorul trebuie sa isi creeze mai intai pe PC-ul propriu o pereche de chei, dintre care doar cheia sa publica va fi cunoscuta de serverul `sshd`. Sintaxa minimala a comenzii de invocare a `ssh` este:

`$ ssh nume_cont@nume_server`

Descrierea detaliata a clientului `ssh` se gaseste in manualul Linux (se afla cu comanda `man ssh`).

◇ PuTTY (http://www.chiark.greenend.org.uk/~sgtatham/putty/)

Este un client ssh utilizat in special pe platforma Windows, cu interfata grafica.

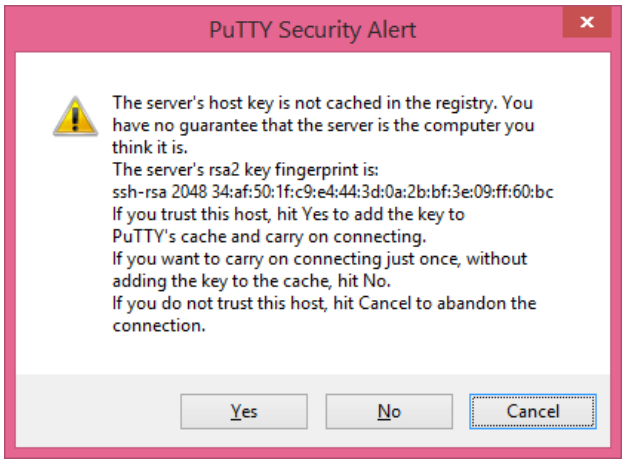


Pentru accesarea unui cont *nume_cont* gazduit pe serverul *nume_server* se face mai intai dublu click pe icoana *putty.exe*.

Deoarece originea software-ului a fost verificata, se ignora mesajul "*Open File – Security Warning*" si se apasa pe butonul "*Run*" pentru a deschide fereastra "*PuTTY Configuration*" reprodusa mai sus.

In continuare se completeaza campul "*Host Name (or IP address)*" cu `nume_cont@nume_server` (de ex. `corina@dirac-client.nipne.ro`) si se apasa pe butonul "Open".

La prima accesare prin clientul respectiv PuTTY a serverului `nume_server` se afiseaza un mesaj prin care utilizatorul este atentionat ca cheia publica a serverului nu este inregistrata in lista de chei a clientului, si se solicita confirmarea utilizatorului privind identitatea serverului.



Daca raspunsul utilizatorului este afirmativ se deschide o interfata in linie de comanda (CLI – *command line interface*) prin care se solicita introducerea parolei, de ex.:

```
Using username "corina".
corina@dirac-client.nipne.ro's password:
```

Obs.: Software-ul descarcat din surse nesigure poate fi periculos. Exemplu mai jos.

Alert: Trojan version of PuTTY has been reported
Fri May 22 10:50:03 EDT 2015
This item has been posted to rhic-rcf-l@lists.bnl.gov, usatlas-users-l@lists.bnl.gov
To all PuTTY and potential PuTTY users of the RACF,
It has been reported that there is a trojanized version of PuTTY being distributed by some sites on the Internet. This version has the capability of stealing the credentials of the user running it. Evidently this trojanized version was originally released in 2013 but it appears it is out in the wild again.
If you are planning on installing PuTTY or upgrading your existing version please use the official site: http://www.chiark.greenend.org.uk/~sgtatham/putty/ or one of its official mirror sites listed there. Don't forget to verify any files you download with the checksums provided.
FYI - http://www.puttyssh.org is an official mirror site but http://www.putty.org is not.
For more information, I found http://it.slashdot.org/story/15/05/19/1422241/trojanized-info-stealing-putty-version-lurking-online to give some useful information including how to check your version. John M. (mccarthy@bnl.gov)

◇ SCP (https://blogs.oracle.com/janp/entry/how_the_scp_protocol_works , 2007)

Secure Copy este un protocol pentru copierea securizata a fisierelor si directoarelor intre masini diferite, folosind ssh pentru transferul de date, autentificare si criptare.

In Linux protocolul este implementat pentru client prin utilitarul `scp`. Sintaxa minimala a comenzii prin care se invoca clientul este:

```
$ scp fisier1 fisier2 ... sursa destinatia
```

Exemple de copiere a fisierului `f00`, respectiv a directorului `d00`, sunt date in tabelul urmator, unde

dst, *dst2* = adresele internet complete ale unor servere nonlocale (ex. `www.nipne.ro`);
cnt, *cnt2* = nume de cont de pe servere nonlocale
/dir, */dir2* = path-uri de directoare

Sursa fisier / director	Destinatie fisier / director	Comanda
Contul *cnt* de pe *dst*	Director local */dir*	`scp cont@dist:f00 /dir`
Contul local	*/dir* din *cont* de pe *dist*	`scp f00 cnt@dist:/dir`
Contul local	*/dir* din *cont* de pe *dist*	`scp -r d00 cnt@dist:/dir`

/dir din *cnt* de pe *dst*	*/dir2* din *cnt2* de pe *dst2*	scp *cnt*@*dst*:/*dir*/f00 *cnt2*@dst2:/*dir2*
Contul local	Dir. radacina pe *cnt@dist*	scp f00 f01 *cnt@dist*:~
/dir din *cnt* de pe *dst*	Directorul curent local	scp *cnt*@*dst*:/*dir*/\{f00,f01\}

◇ WinSCP (http://winscp.net)

Este versiunea clientului *scp* pentru platforma Windows, cu interfata si procedura de acces asemanatoare cu cea a clientului *putty*.

Fereastra de navigare este impartita in doua, o parte pentru directorul curent al PC-ului local iar cealalta pentru directorul curent al serverului, astfel incat transferul de fisiere/directoare se poate face prin *drag-and-drop*.

WinSCP dispune de numeroase facilitati, printre care posibilitatea de editare a fisierelor text si setarea drepturilor de acces.

Pentru setarea drepturilor de acces ale unui fisier/director se face click-dreapta pe icoana respectiva si se selecteaza "*Properties*".

❖ ANEXA 2: CATEVA NOTIUNI SI COMENZI LINUX

In continuare sunt prezentate sumar doar informatiile necesare pentru acest curs. Un text mai explicit se gaseste, de exemplu, in [W. Shotts, *The Linux Command Line*, 2013, http://linuxcommand.org].

◇ A2.1 Sistemul de fisiere Linux

Sistem de fisiere = modul de organizare al fisierelor si directoarelor intr-un sistem de operare.

Directorul = un fisier care este o lista de alte fisiere

In sistemul de fisiere Linux directoarele sunt dispuse intr-o structura ierarhica (arbore), pornind de la directorul radacina al sistemului, `/`.

Dupa autorizarea accesului la contul `nume_cont` (utilizand PuTTY sau altfel), utilizatorul se afla in directorul care i-a fost alocat de catre administrator, numit in continuare "*directorul radacina al utilizatorului*" (`/home/`*`nume_cont`*, pe scurt `~`). Numele directorului curent se poate verifica cu comanda "*print working directory*"

```
$ pwd
```

care are ca rezultat afisarea caii absolute a acestuia in structura ierarhica de fisiere (in acest caz `/home/`*`nume_cont`*, unde `home` este un subdirector al lui `/` iar *`nume_cont`* un subdirector al lui `home`).

Obs.: Descrierea completa a oricarei comenzi Linux se obtine prin

```
$ man nume_comanda
```

Aici nu va fi prezentata decat sintaxa minimala a comenzilor strict necesara pentru curs.

Daca intr-o comanda nu se specifica calea (*path*) relativa la pozitia curenta unui director/fisier, sistemul aplica comanda directorului curent. De exemplu, calea absoluta catre fisierul `fisier` aflat in directorul radacina al utilizatorului, care este `/home/`*`nume_cont`*`/fisier`, pe scurt `~/fisier`, nu mai trebuie sa fie specificata daca utilizatorul se afla in `~`.

In continuare se va considera ca numele unui director/fisier include calea, atunci cand este necesara specificarea acesteia.

◇ A2.2 Lucrul cu fisiere si directoare

Pentru a lista continutul directorului *`nume_director`* se da comanda

```
$ ls –al nume_director
```

unde prin "`a`" se solicita listarea *tuturor* fisierelor/subdirectoarelor, iar prin "`l`" se solicita informatii detaliate despre acestea. De exemplu:

```
[mdulea@dirac-client ~]$ ls -al
total 60
drwx------. 8 mdulea users 4096 Jun 29 13:52 .
drwxr-xr-x. 6 root   root  4096 Jul 10 10:53 ..
-rw-------. 1 mdulea users 3568 Jul  9 22:26 .bash_history
-rw-r--r--. 1 mdulea users   18 Oct 14  2014 .bash_logout
-rw-r--r--. 1 mdulea users  176 Oct 14  2014 .bash_profile
-rw-r--r--. 1 mdulea users  151 Jun 10 15:05 .bashrc
drwxrwxr-x. 8 mdulea users 4096 Jun 10 15:07 dirac-client
drwxrwxr-x. 2 mdulea users 4096 Jun 29 08:54 .globus
drwxr-xr-x. 2 mdulea users 4096 Nov 23  2010 .gnome2
drwxrwxr-x. 2 mdulea users 4096 Jun 10 14:48 .installCache
drwxrwxr-x  4 mdulea users 4096 Jun 29 13:27 jobs
drwxr-xr-x. 4 mdulea users 4096 Feb  6  2014 .mozilla
-rw-------. 1 mdulea users 1024 Jun 10 12:59 .rnd
-rw-------  1 mdulea users 2864 Jun 29 13:52 .viminfo
-rw-------. 1 mdulea users   67 Jun 10 11:13 .Xauthority
[mdulea@dirac-client ~]$
```

Fiecare linie afisata descrie un fisier sau subdirector din directorul respectiv. Primul caracter din prima coloana se refera la tipul entitatii (`d`=director, `-`=fisier). Urmatoarele 9 caractere specifica drepturile de acces la entitate pentru detinatorul contului (caracterele 2-4), pentru grupul de care apartine detinatorul contului (caracterele 5-7) si, respectiv, pentru ceilalti utilizatori (caracterele 8-10). Valorile pot fi `r` = citire (*read*), `w` = scriere (*write*), `x` = executie (*execute*), sau - = drept interzis.

Numele de cont al detinatorului este listat in Coloana 3, iar numele grupului sau in coloana 4. Numele fisierului/directorului este specificat in ultima coloana.

Sintaxa simplificata a comenzii prin care detinatorul fisierului *`nume_fisier`* modifica drepturile de acces la acesta este urmatoarea [http://ss64.com/bash/chmod.html] DIRECTOR?

```
$ chmod kmn nume_fisier
```

unde corespondenta dintre valorile luate de oricare dintre numerele intregi *`k`*, *`m`*, *`n`* si drepturile corespunzatoare de acces ale detinatorului (`u`=*user*), ale grupului (`g`=*group*), respectiv ale celorlalti utilizatori (`o`=*others*) este data in tabelul urmator:

---	--x	-w-	-wx	r--	r-x	rw-	rwx
0	1	2	3	4	5	6	7

De exemplu stabilirea drepturilor de acces la fisierul `nume_fisier` de citire, scriere si executie (`rwx`) pentru detinator (`u`), citire si executie (`r-x`) pentru grup (`g`) si, respectiv, interzicerea accesului (`---`) pentru ceilalti utilizatori (o), se face prin comanda:

```
$ chmod 750 nume_fisier
```

Sau, echivalent, cu

```
$ chmod u=rwx,g=rx nume_fisier
```

`chmod` actioneaza asemanator asupra directoarelor, dar in acest caz semnificatia permisiunilor `r`, `w`, `x` este urmatoarea: `r` = se pot lista fisierele din director; `w` = se pot adauga fisiere noi in director; `x` = se pot accesa fisierele din director.

Trecerea de la un director la altul se face cu comanda

```
$ cd nume_director_nou
```

calea specificata putand fi cea relativa sau absoluta.

Pentru crearea directorului *nume_director* (de ex. `grid`) se foloseste comanda

```
$ mkdir nume_director
```

Fisierele si directoarele pot fi sterse (**definitiv !**) cu comanda `rm`:

```
$ rm nume_fisier
$ rm -r nume_director
```

Fisierele si directoarele pot fi copiate cu comanda `cp`:

```
$ cp nume_fisier_de_copiat nume_fisier_rezultant
```

Fisierele si directoarele pot fi mutate sau redenumite cu comanda `mv`:

```
$ mv nume_fisier_de_mutat nume_director_tinta
$ mv nume_vechi_fisier nume_nou_fisier
```

◇ A2.3 Vizualizarea continutului fisierelor

Afisarea integrala a continutului unui fisier se face cu `cat`

```
$ cat nume_fisier
```

Pentru navigarea linie cu linie intr-un fisier se utilizeaza `less`

```
$ less nume_fisier
```

Afisarea primelor n linii dintr-un fisier se obtine cu `head`

```
$ head –n nume_fisier
```

Afisarea ultimelor n linii dintr-un fisier se obtine cu `tail`

```
$ tail –n nume_fisier
```

❖ ANEXA 3: NOTIUNI DE UTILIZARE A EDITORULUI VIM

VIM este un editor *modal* in linie de comanda inclus in majoritatea distributiilor Linux.

Comanda de editare a unui fisier nou sau existent este

```
$ vim nume_fisier
```

Din acest moment editorul se afla in modul NORMAL, in care se poate naviga prin fisier.

Navigarea caracter cu caracter in modul NORMAL se face folosind randul de taste "h" (stanga), "j" (jos), "k" (sus) si "l" (dreapta). Pentru deplasari pe distante mai mari se folosesc tastele:

"^" – inceput de rand

"$" – sfarsit de rand

"gg" – primul rand

"G" – ultimul rand

Pe langa modul NORMAL, editorul mai poate fi in alte 4 moduri; informatii privind modul curent sunt afisate in partea stanga a barei inferioare.

Trecerea din modul normal in modul INSERT, in care se editeaza textul, se face apasand tasta "i". Din acest moment se poate insera un text prin tastare directa sau prin copiere.

In modul INSERT tastele au functiile obisnuite, dar mouse-ul nu poate fi folosit pentru deplasare. Pentru copierea unui text dintr-un document editat pe PC in Windows, se face mai intai "copy" cu mouse-ul sau Ctrl+c, iar apoi se face click dreapta in fisierul deschis in modul INSERT. Textul copiat va fi inserat in fisier inaintea pozitiei cursorului.

Pentru salvarea fisierului editat se iese mai intai din modul INSERT in modul NORMAL apasand tasta "Esc"; in partea stanga a barei inferioare apare cuvantul "*recording*".

Se tasteaza apoi ":wq", urmat de "Enter". Explicatie: prin tastarea ":" s-a trecut din modul NORMAL in modul COMANDA; "w" (*write*) este comanda de salvare a modificarilor efectuate, iar "q" (*quit*) determina iesirea din editor.

❖ ANEXA 4: JOB DESCRIPTION LANGUAGE (JDL)

(O descriere detaliata a JDL se gaseste la https://wiki.italiangrid.it/twiki/bin/view/CREAM/JdlGuide)

JDL este un limbaj de nivel inalt folosit de catre utilizatorii grid pentru a comunica sistemului informatiile necesare pentru rularea job-urilor (numele executabilului, sursa datelor de intrare, cerintele de resurse, instructiuni de rulare, modul de prezentare/recuperare al datelor de iesire, etc.). Aceste informatii sunt prezentate in fisiere cu extensia `.jdl`, care contin linii de text in formatul

```
[
atribut1 = expresie1;
```

***atribut*$_2$ = *expresie*$_2$;**

...

***atribut*$_n$ = *expresie*$_n$**

]

Sirurile de caractere se incadreaza cu " ". Caracterul " se desemneaza intr-un sir de caractere prin \"

Liniile commentate incep cu **#** sau **//** ; pentru mai multe linii comentate se procedeaza ca in limbajul C.

Atributul **Executable** specifica numele fisierului de comenzi ce vor fi executate pe un WN.

Daca fisierul nu este deja pe WN, ci trebuie mai intai sa fie copiat de pe UI, se adauga linia:

InputSandbox = {"/home/*nume_cont_utilizator*/*fisier_de comenzi*"};

Atributele **StdOutput** si **StdError** specifica numele fisierelor cu rezultate, respectiv erori.