'E pur si muove'

Jocuri sub Linux

Review OneBase Linux

Arta documentării software-ului

Programarea şi uneltele ei

linux360

septembrie 2004

09

.coperta

Cine sunt eu?

Visez la o lume liberă. Liberă de constrângeri, de limite, de refuzuri. Vreau să nu mi se spună ce să îmbrac şi ce să mănânc. Să nu mai decidă oameni şi maşini în locul meu. Am obosit. Uitați-vă într-o zi în jur. Veți vedea sisteme de securitate pe care le-ați luat drept naturale până acum. Se vor ivi privirii arme proiectate nu pentru a servi în luptă, ci pentru a menține frica. Se vor dezveli mecanisme gândite să vă țină într-un sistem de legi fără să fiți conştienți. Începe să sune cunoscut?

Am găsit mai de mult o insulă de libertate. Nu în ceea ce-l priveşte pe om, acesta fiind avid de putere şi control pentru multe milenii ce au să vină. Aliatul l-am găsit, o ironie a sorții, în lumea maşinilor - calculatorul. Acesta mi-a arătat o lume în care oamenii împart idei, nu şi le impun, colaborează, nu comandă, tind spre un ideal comun, nu muncesc pentru binele firmei. O lume a software-ului liber.

„Şi ce, eu nu pot alege dacă să scriu în Lotus sau în Word?" ar întreba funcționarul mediocru de la minister. „Desigur, monşer", i-ar răspunde Mitică. „Tu şi alte 22 de milioane de oi".

Mi-a fost frică la început. Că nu voi putea face tot ce puteam face înainte. Că mă voi lovi de probleme cărora sigur nu le voi face față. Sau că, mai rău, voi pierde documente. Răul, deşi aşteptat, nu s-a întâmplat. Am găsit alți oameni liberi, ca şi mine. Care m-au ajutat când am întrebat şi m-au ghidat prin întunericul noutății. Şi, uşor uşor, n-am mai depins de nimic din cele vechi.

Acum încerc să aduc o contribuție şi să le ofer celor ce merită o mică privire în interiorul acestei lumi. Vor avea posibilitatea unei alegeri. Şi nu pot să sper decât că vor face aceeaşi alegere ca şi mine.

Ovidiu

linux360 - nr. 09 septembrie 2004

www.linux360.ro / redactia@linux360.ro

Echipa

Ovidiu Lixandru - director general
Răzvan Şocu - director general
Radu Mihăilescu - redactor-şef
Ciprian Negrilă - redactor
Cristian Bidea - redactor
Dan Marcu - redactor
Daniel Secăreanu - redactor
Ioana Gliția - redactor
Anca Holban - colaborator
Costin Gamenț - colaborator
Alex Bucur - colaborator
Abibula Aygun - colaborator

Interviu cu dezvoltatorii Ampache

Radu - Eosif Mihăilescu

Poate că mulți dintre noi sunt melomani şi poate că mulți dintre noi au în posesie o colecție de muzică în format electronic - mai mare sau mai mică, mai specifică sau mai variată, în format `.MP3` sau `.OGG` dar în orice caz foarte îndragită şi foarte ascultată de posesor. Orice colecție, de orice natură ar fi ea, are nevoie de o formă de organizare (indexare) pentru a putea (mai ales în cazul colecțiilor mari) căuta şi regăsi comod diverse obiecte stocate în ea.

Eu personal am acces la o colecție muzicală ce măsoară în jur de 7800 de piese, toate în format `.MP3`, aşa că am tot căutat o soluție de indexare care să fie uşor de instalat şi configurat, care să facă față dimensiunilor colecției în cauză şi care să treacă neobservată (din punctul de vedere al resurselor consumate) pe server-ul pe care urma să fie instalată.

După experiențe neplăcute cu "*soluții*" din cele mai diverse, de la **Andromeda** la `mod_musicindex`, am descoperit Ampache... şi Ampache a fost. În acest articol vă voi prezenta pe Ampache, un veritabil "*tonomat*" pentru web dar nu înainte de a vă prezenta un interviu cu doi dintre dezvoltatori.

Interviul a avut loc în paralel, având ca protagonişti pe Andy Morgan (programator) şi pe Karl Vollmer (conducătorul proiectului).

linux360: Ampache este un proiect ce are legătură cu Linux. Când s-a "întâlnit" echpa Ampache prima oară cu Linux-ul?

Andy Morgan: *Am început să mă joc cu Linux în 1996. Configuram un server de web pentru prima oară folosind Slackware. La puțin timp după aceea am început să întrețin server-ul de web al "Student Computing Facilities" (ca student angajat) aici la OSU [Oregon State University, n. ed.].*

Karl Vollmer: *Am făcut prima oară cunoştință cu Linux-ul acum aproape cinci ani într-o încercare de a găsi o alternativă liberă la Windows şi Office. De atunci am început să-l folosesc ca principal sistem de operare la muncă şi acasă. Nu pot trăi fără Linux :-)*

l360: Ce v-a făcut să-l creați pe Ampache? Care a fost ideea inițială de la care ați început?

A.M.: *Nu eu am început pe Ampache. Am început să scriu cod pentru el după ce l-am folosit o vreme.*

K.V.: *Ei bine, nici eu nu am început pe Ampache - a fost creat de Scott Kveton. L-am întrebat pe Kveton şi a spus că motivul era că lucra în două birouri diferite şi se plictisise să-şi tot mute muzica dintr-o parte în alta. La acea vreme nu a găsit nici o altă aplicație care putea să facă ce face Ampache.*

l360: De ce ați ales platforma curentă (PHP/MySQL/Apache) pentru Ampache în loc de a-l dezvolta folosind una dintre infrastructurile alternative existente astăzi (e.g ASP, JSP etc.)?

A.M.: *Vezi răspunsul meu anterior - deşi un răspuns scurt ar fi "open source". LAMP (Linux, Apache, MySQL, PHP) este calea de urmat pentru aplicații web ca aceasta.*

K.V.: *M-am decis să rămân la PHP/MySQL în parte pentru că asta ştiu şi în parte pentru că rulează pe majoritatea platformelor. De curând am adăugat câteva script-uri pentru linia de comandă care sunt scrise în Perl. Aş recomanda cu certitudine acelaşi set de unelte oricui s-ar apuca de un proiect asemănător lui Ampache. Sunt uşor de învățat, uşor de folosit şi duc treaba la bun sfârşit.*

l360: De ce ați ales să scrieți Ampache când există deja alte proiecte Open Source servind acelaşi scop?

A.M.: *Nu cred ca că existau proiecte similare în momentul când Ampache a fost început.*

K.V.: *Motivele pentru care am contribuit la (ca mai apoi să devin principalul dezvoltator al) Ampache sunt pur egoiste. În timp ce creştea, colecția mea de MP3-uri devenea din ce în ce mai greu de întreținut. Ampache rezolvă această problemă şi, ca să fiu cinstit, nici să primeşti bere pe gratis [să ți se facă cinste, n. tr.] facând ceva distractiv nu e tocmai rău. Cât despre continuarea dezvoltării Ampache când sunt alte tonomate pentru web în jur, cred că designul şi modul de organizare al informației în Ampache sunt unice şi, din câte ştiu, antedatează majoritatea tonomatelor de web existente.*

l360: Cât de departe în trecut merge Ampache (când v-ați apucat de el)?

A.M.: *2001 cred, dar Vollmer s-ar putea să ştie sigur.*

K.V.: *Am început să lucrez la el la scurt timp după ce l-am întâlnit pe Scott Kveton, fondatorul Ampache. Abia în Aprilie 2003 am preluat conducerea proiectului deoarece Scott Kveton pur şi simplu nu mai avea timp să lucreze la el. Ampache ca atare a fost prima oară lansat pe 29 Aprilie 2001.*

I360: Care a fost partea cea mai dificilă din Ampache – mai întâi din punct de vedere al designului şi apoi din cel al implementării/programării?

A.M.: *Pentru mine, a transforma Ampache de la o implementare cu un fişier de date şi o bibliotecă de funcţii la un design orientat pe obiecte a constituit volumul cel mai mare de muncă. Ne-am gândit mult si bine ce clase să creăm şi care funcţii trebuie să fie în ce clase. Majoritatea programării este uşoară, dar munca de design cere o planificare atentă şi gândire proactivă.*

K.V.: *Designul noului cod orientat pe obiecte menţinând în acelaşi timp compatibilitatea pe verticală trebuie că a fost cea mai grea parte. Când am început să lucrez la Ampache, aproape orice schimbare de design făceam necesita schimbări masive în mai multe funcţii diferite din cod. Pe măsură ce am migrat spre un design orientat pe obiecte, facilităţile noi şi corecturile au devenit din ce în ce mai uşor de implementat. Aş spune că singura şi cea mai importantă parte a oricărui proiect este crearea unui design uşor extensibil.*

I360: Îl vedeţi pe Ampache existând şi anul viitor? Dar peste 5 ani?

A.M.: *Ampache serveşte perfect nevoilor mele, deci eu (personal) nu am nici un motiv să folosesc un alt produs. Dacă va exista vreo facilitate de care voi avea nevoie în viitor, o voi adăuga la Ampache :). O îngrijorare o constituie totuşi legile americane de tipul DMCA-ului [Digital Millenium Copyright Act, n. ed.]. Este de conceput că Ampache ar putea deveni ilegal [în acest context], chiar dacă eu cred că [acest lucru] este improbabil.*

K.V.: *Dacă depinde de mine, sigur. Sper să continui să-l dezvolt pe Ampache pentru mulţi ani de acum înainte. Sigur, au fost câteva temeri pe marginea DMCA şi a altor legi asemănătoare ce ar face programe ca Ampache ilegale, dar cred că prin obligativitatea tranzacţiei de autentificare cu parolă ne aducem contribuţia la prevenirea distribuirii ilicite [de muzică].*

I360: Care sunt planurile dumneavoastră (dacă există) pentru Ampache? Ce este cel mai probabil să fie adăugat şi/sau scos şi/sau schimbat în versiunile următoare?

A.M.: *Cred că facilităţile care vor fi adăugate curând (multicasting şi editarea repertoriului în timpul redării) vor permite lui Ampache să se comporte şi ca o soluţie de streaming gen "post de radio". Aceasta îl va face pe Ampache mai util pentru mai mulţi oameni.*

K.V.: *Mi-ar place foarte mult să-l fac pe Ampache mai asemănător cu un post de radio [digital], permiţând astfel cereri [pentru o piesă] şi multicasting. De asemenea mi-ar place foarte mult să găsesc o cale de a-l face acceptabil într-un mediu comercial. Codul pentru blocarea pieselor, care permite redarea doar a unei singure copii a unei melodii la un moment dat, este primul pas către atingerea acestui scop. Totuşi, am o cerinţă pentru orice modificări care restricţionează redarea/descărcarea de muzică din Ampache: trebuie să le poţi dezactiva!*

I360: Întrebare finală: ca membri ai unui proiect Open Source, ce aveţi de zis oamenilor care contemplează ideea de a porni un astfel de proiect pe cont propriu sau de a contribui la unul existent? Ar trebui să-şi urmeze ideea? Ce credeţi că este cel mai important lucru pe/de care ar trebui să-l ştie/să fie conştienţi înainte de a începe?

A.M.: *Este important să ai o aplicaţie (în mare parte) funcţională pentru a putea obţine ajutorul dezvoltatorilor din afară. Oamenilor le place să lucreze la proiecte existente pentru a adăuga facilităţi noi şi modificări, dar preferă să aibă ceva funcţional cu care să se joace la început. Nu vei obţine prea mult ajutor la dezvoltare doar creând un arbore CVS vid la Sourceforge. :)*

K.V.: *Sunt puţine lucruri care cred eu că ajută proiectele Open Source să aibă succes. În primul rând, am continuat să lucrăm la el. Chiar dacă a fost un hiatus după ce Kveton mi-a transferat conducerea, nu l-am lăsat să moară. În al doilea rând, trebuie să oferi "suport" bun pentru aplicaţia ta. M-am îndepărtat de câteva proiecte doar din cauză că dezvoltatorii lor erau absenţi sau nepoliticoşi. Dacă începi un proiect pe cont propriu, nu-l face public înaine de a avea un produs funcţional. S-ar putea să pierzi mulţi utilizatori iniţiali dacă produsul tău nu funcţionează de la prima încercare. Dacă ai intenţia să contribui la un proiect, prima şi prima oară conformează-te standardelor de redactare a codului existente. Nimic nu enervează mai mult un dezvoltator decât să primească un patch care este redactat într-un stil complet diferit de restul codului său. În ultimul rând, comentează-ţi codul, facând prin aceasta uşor pentru ceilalţi oameni să-şi dea seama ce naiba faci [acolo]!*

Pagina de login

Aici ia sfârşit interviul cu Andy Morgan si Karl Vollmer, dezvoltatorii principali ai proiectului Ampache.

Vom reveni în numărul următor cu o prezentare a produsului propriu-zis folosindu-ne de o instalare aflată "în producţie" de aproape un an de zile.

Resurse:

- http://www.ampache.org/

Autor:

radu.mihailescu@linux360.ro

LinuxFest Cluj-Napoca

Radu - Eosif Mihăilescu

Poate unii dintre Dumneavoastră sunt familiari cu termenul “LinuxFest”, mai ales din literatura de specialitate şi din grupajele de ştiri din domeniu. Pentru aceia care nu sunt, vom aduce puțină lumină asupra noțiunii în cele ce urmează.

Un LinuxFest (“*petrecere de Linux*” în traducere mot-á-mot) este o manifestare organizată de un grup de entuziaşti ai domeniului pentru a putea discuta aspecte tehnice, a se cunoaşte mai bine şi chiar a face prezentări de profil şi/sau a-i învăța şi pe alții. În cazul în care tematica unui LinuxFest este, în mod explicit, instalarea unei distribuții drept exemplu practic, el poartă denumirea de InstallFest.

Spuneam că, probabil, ați auzit de LinuxFest-uri din presa externă şi asta pentru că acest “fenomen” a luat amploare abia de curând la noi în țară. Pentru a ajunge la tema prezentului articol, trebuie să vă spun că, în urma mişcărilor active de acest gen din capitală, în luna martie a acestui an (mai precis pe 20) a avut loc în Cluj-Napoca un prim eveniment de acest gen. Domnul redactor linux360 Florin Vereş, din Cluj-Napoca, a avut amabilitatea să ne răspundă la câteva întrebări despre evenimentul al cărui organizator a fost:

linux360: Dumneavoastră şi Linux-ul.... cum ați ajuns "împreună"?

Florin Vereş: *Prima mea întâlnire cu Linux-ul a fost în anul 2000, când unul din colegii mai mari mi-a imprumutat un CD cu RedHat 6.2 (eu eram pe clasa a VI-a, el era pe a XII-a). A fost "dragoste" la prima vedere, dar, din păcate, nu am putut să-l folosesc zilnic deoarece pe acel sistem se făcea tehnoredactare, şi nu aveam un hard-disk suficient de încăpător să am pe el dual-boot.*

l360: Ce vă place cel mai mult în/la el?

F.V.: *Îmi place foarte mult ideea de Open Source, dar, de asemenea, îmi place să ştiu exact ce se întamplă cu calculatorul meu.*

l360: De unde ideea unui LinuxFest la Cluj?

F.V.: *Ideea mi-a venit în urma unei discuții pe forumul linux360 referitoare la LinuxFest-ul din Bucureşti. M-am gândit că din moment ce este la Bucureşti aşa ceva, de ce să nu fie şi în Cluj?*

l360: De unde a venit suportul (oameni/logistică/idei) - dacă a venit? Dacă nu a venit, cum ați reuşit totuşi?

F.V.: *Din păcate, logistică nu am prea avut. Dar ne-am străduit să aducem oameni, făcându-i reclamă LinuxFest-ului pe listele de discuții şi forumuri de Linux şi nu numai (e. g. linux360, rlug, MandrakeNation, Chip etc.).*

l360: Odată hotărâtă organizarea manifestării, cum ați descrie/caracteriza reacția celor anunțați/interesați?

F.V.: *Cei anunțați au fost foarte interesați, mai ales că a fost prima întâlnire de acest gen din Cluj-Napoca.*

l360: V-ați aşteptat să aveți aşa o participare numeroasă chiar de la prima ediție?

F.V.: *Recunosc că eu mă aşteptam să vină 10-15 oameni, dar am fost plăcut surprins când am văzut că au participat 34 de oameni.*

l360: Ce s-a întâmplat de ați avut tot, mai puțin sală?

F.V.: *Din păcate, nu am putut face rost de sală. Decanul de la Poli [Politehnica din Cluj, n. ed.] a fost de acord să ne dea sala, dar până la urmă, din diverse motive, nu ne-a putut ajuta.*

l360: Cum ați caracteriza desfăşurarea primei ediții (din punct de vedere pur ştiințific)?

F.V.: *Din punct de vedere ştiințific, s-a desfăşurat destul de bine. Nu au fost aşa-numitele "distro wars", dar a fost şi o discuție referitoare la distribuții, dar doar din punct de vedere a avantajelor diferitelor distribuții importante. Un alt subiect mult discutat a fost Networking, care este prezent pretutindeni în lumea Linux.*

l360: Cum ați caracteriza desfăşurarea primei ediții (din punct de vedere al participanților ca oameni)?

F.V.: *Din punct de vedere al participanților a fost OK. Nu au fost prezenți numai "Linux gurus", ci şi oameni care nu ştiau nimic despre Linux, sau ştiau doar foarte puțin, dar doreau o alternativă. Menționez că au fost participanți nu numai din Cluj-Napoca, ci şi din Alba Iulia, Şimleul Silvaniei, dar şi din Republica Moldova.*

l360: Primul a fost. Vor mai fi LinuxFest-uri la Cluj?

F.V.: *Noi facem totul posibil să mai facem LinuxFest-uri. Poate că nu săptămânal ca şi în Bucureşti, ci doar o dată la 1-2 luni, deoarece Clujul nu e aşa mare ca Bucureştiul.*

Autor:

radu.mihailescu@linux360.ro

ROSLIMS - Knoppix a învățat româna

Ovidiu Lixandru

Un sistem de operare se poate numi user-friendly sau, ca să nu supărăm extremiştii în ale lingvisticii, prietenos cu utilizatorul, în momentul în care (şi) interfața sa „vorbeşte” aceeaşi limbă cu cel din fața tastaturii.

„Ştie să vorbeşte?”

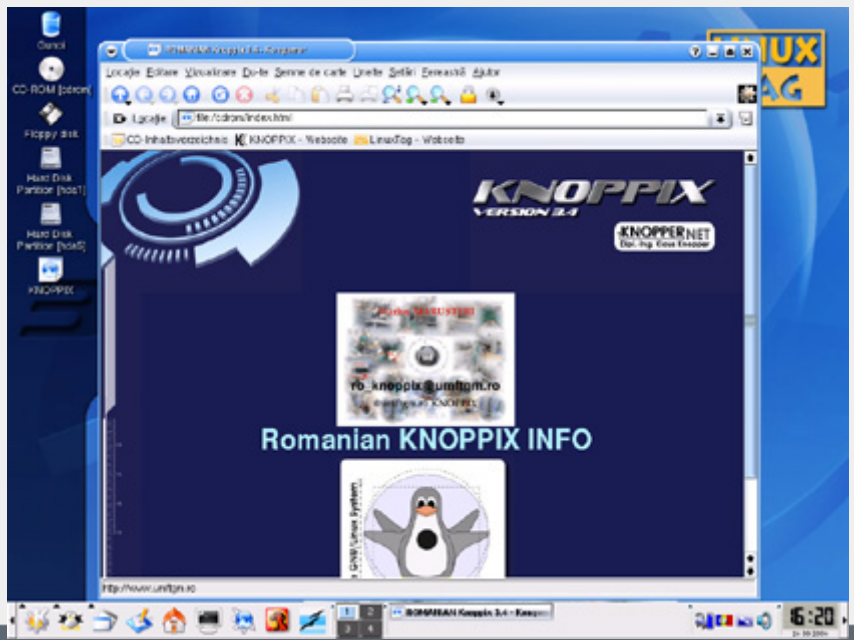
Limba română a fost atacată pe mai multe fronturi Linux, cele mai importante distribuții localizate complet la nivelul interfeței fiind Darkstar, Vision şi ROSLIMS. Dacă primele două mai au ceva de muncă până a ajunge în stadiul de „producție”, ROSLIMS este aici şi acum.

De ROSLIMS nu auzisem nimic până acum câteva săptămâni. Mi-au povestit de el colegii întorşi de la un seminar Linux arădean. Dezvoltat de dr. Marius Măruşteri de la Universitatea de Medicină şi Farmacie din Târgu Mureş, distribuția se numea cândva Romanian Knoppix. Dar, prin adăugarea unor aplicații cu specific medical şi îmbunătățirea suportului pentru limba română, autorului i-a fost cerut de către utilizatori (proprii studenți) să-i fie schimbat numele distribuției deoarece nu mai era un simplu Knoppix.

Nu vă lăsați păcăliți de splash-uri. Artwork-ul e un pic în spatele numelui.

Ce este ROSLIMS?

Romanian Simple Linux for Medical Students poate fi descris ca un Knoppix cu interfața în română şi un pachet de finețuri ce îi aduc un plus de valoare față de fratele german. Un LiveCD deci, numai bun de încercat Linux pentru prima dată sau folosit pe calculatoare cu alte sisteme de de operare instalate şi pe care nu vreți să le „deranjați”.



La plăcinte înainte

Primul lucru ce merită menționat încă de la boot este layout-ul tastaturii ce vine predefinit pe engleză şi nu mai trebuie să te chinuieşti ca la Knoppix să găseşti caracterul =.

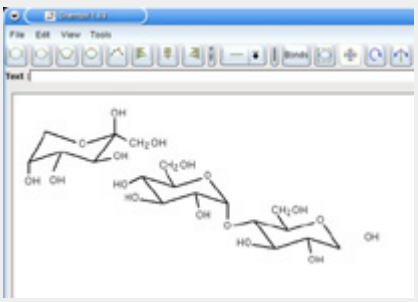
O mică avertizare - chiar dacă vedeți scris pe ici, pe colo, Romanian Knoppix, să nu credeți că aveți o versiune mai veche a distribuției - pur şi simplu timpul nu a permis ca schimbarea numelui să se reflecte şi în grafică.

După un `knoppix lang=ro` şi un `ENTER`, se trece la procesul de autoconfigurare al hardware-ului aflat în calculator. Acesta se descurcă excelent,

de la placa video până la CD-writer-ul IDE.

O surpriză ceva mai puțin plăcută a fost la intrarea în modul grafic, moment în care monitorul a intrat în standby și semn că s-a încercat setarea unui mod în afara celor suportate de el. Nimic nou sub soare, aceeași problemă o avusesem și cu Knoppix. Am rebootat și am recitit instrucțiunile disponibile la boot prin apăsarea tastelor `F2` și `F3`. Astfel, printr-o comandă de boot de forma `knoppix lang=ro screen=800x600 depth=24 vsync=75` i-am indicat manual parametrii video necesari pentru rularea în condiții bune pe sistemul meu.

În sfârșit cu KDE în față, am purces la explorat.



Chemtool

Extra, extra!

La nivelul interfeței, lucrurile se prezintă la fel ca și până acum: tema vizuală Keramik, același aranjament al meniurilor, aceleași aplicații standard. Să vedem totuși care au fost motivele pentru care Romanian Knoppix a devenit ROSLIMS.

„Medical”-ul din nume se datorează celor 3 aplicații cu acest specific prezente pe CD: `Chemtool`, `XDrawChem` și `Pybliographic`. Primele două pot fi folosite pentru manipularea a diverse tipuri de molecule. Ele vin atât cu șabloane, cât și cu instrumente de desenare și editare, putându-se lucra și cu mouse-ul, și cu tastatura. Cea de-a treia aplicație, `Pybliographic`, se ocupă de gestionarea referințelor bibliografice, cu ajutorul său putându-se efectua și interogări Medline - scopul utilizării nu cred că mai trebuie menționat.

Office

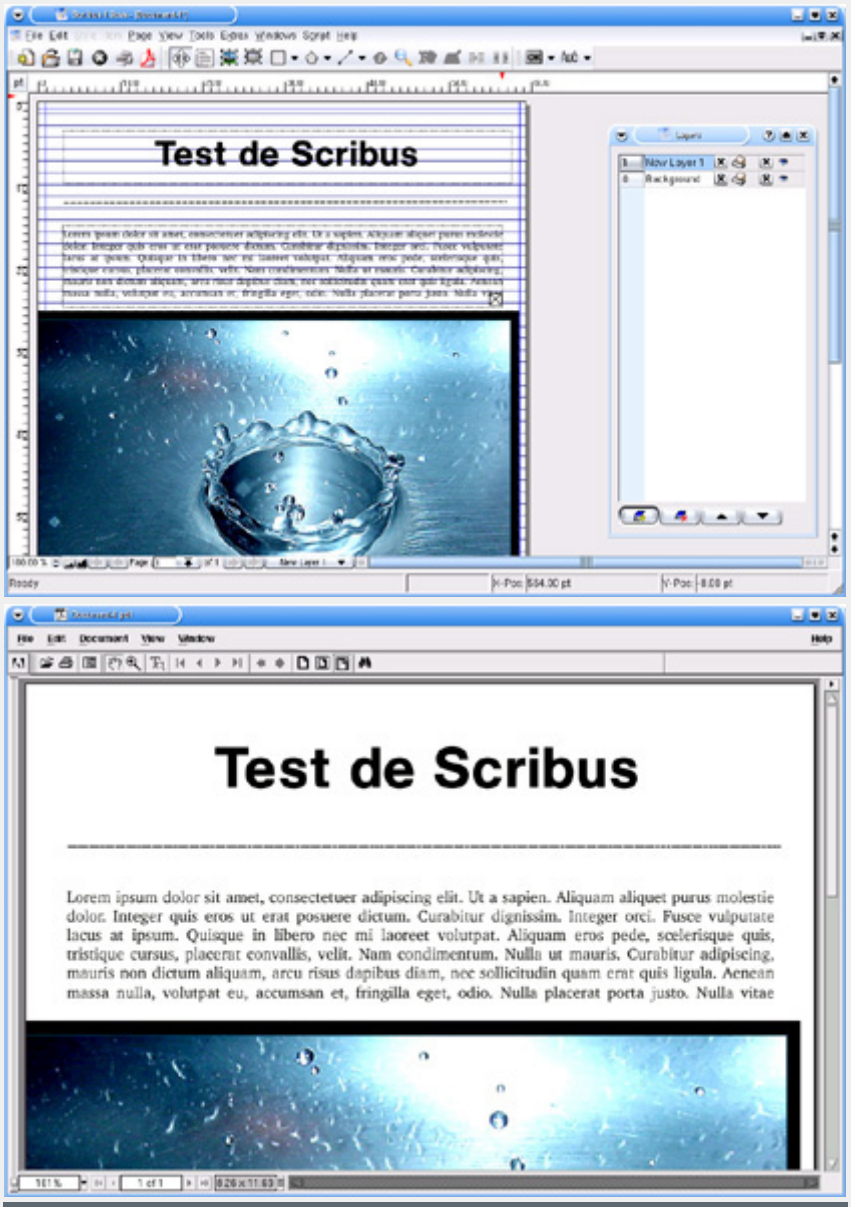
Așa cum și autorul menționează, distribuția poate fi folosită la fel de bine și în alte domenii decât cel medical.

De exemplu, OpenOffice.org, inclus în versiunea 1.1.1, este cu siguranță o unealtă ce va satisface nevoile oricărui birou de firmă medie sau mijlocie. El se ocupă de tot ce înseamnă editare de text, calcul tabelar, prezentări și multe altele. Dacă nu sunteți familiarizat cu această suită office, puteți găsi într-un număr anterior al linux360 o prezentare în detaliu a acesteia.

O altă facilitate excelentă pentru un

Server de print PDF

birou este aceea de creare a unui print-server PDF cu doar câteva click-uri ce va fi vizibil în întreaga rețea, fie ea Linux sau



Sus - un layout în Scribus; jos - exportul în PDF. Remarcați calitatea.

Windows. Obțineți astfel o funționalitate ce, cu ajutorul unui soft comercial, poate implica o investiție de câteva sute de euro.

DTP

Ați avut vreodată ceva de tipărit la o calitate cel puțin semi-profesională? Atunci ştiți cu siguranță că un fişier generat de un editor de text nu este deloc recomandat, ci de o aplicație dedicată. ROSLIMS include o unealtă din domeniul Adobe InDesign şi Quark Xpress pentru desktop publishing - Scribus 1.2. Deşi relativ tânără, aplicația „musteşte" de bunătăți ce pot fi regăsite numai pe bani grei la cele două programe comerciale antemenționate. Pe lângă suportul foarte bun de layere, formatare a textului şi a imeginii, Scribus se bucură de un suport de exportare a documentelor create în format PDF la rezoluții variabile, de la obişnuitul 72dpi pentru ecran până la câteva mii de dpi pentru print.

CAD

ROSLIMS poate fi folosit şi în domeniul Computer Aided Design prin intermediul lui QCad. Acesta este, la fel ca şi Scribus, din zona semi-profesională şi se adresează utilizatorlor începători şi medii ai acestui tip de programe.

Multimedia

ROSLIMS are aplicații interesante şi pentru PC-ul de acasă. De la player-ul de muzică XMMS şi până la cel video Xine, acestea nu vor avea probleme cu nici un format de fişiere, codec-urile necesare redării fiind incluse în pachet.

Xine şi ATB

Pasionații muzicii vor regăsi, pe lângă un editor de fişiere audio foarte performant, Audacity, şi un editor de partituri muzicale, Rosegarden. Placa PC-TV nu va fi nici ea lăsată pe dinafară, XawTV fiind la datorie. Jocurile sunt prezente cu duimul (categoria mici şi distractive) şi vor face cu siguranță deliciul copiilor dvs sau, de ce nu, chiar al părinților.

Rețelistică

ROSLIMS are posibilitatea de a fi configurat, prin intermediul unui wizard, şi ca gateway pentru un birou sau acasă, unde există mai multe calculatoare şi o singură legătură la Internet. Veți obține astfel un router foarte sigur, chiar şi în cazul compromiterii nefiind nevoie decât de un reboot şi o reconfigurare. Natura read-only a mediului de pe care rulează (CD) îi conferă această proprietate, neputându-se şterge sau modifica nimic.

Sistem

Posibilitățile de backup pentru diverse sisteme şi tipuri de partiții ce există în Knoppix s-au păstrat şi aici. Ion Mudreac are un articol dedicat acestei teme într-unul din numerele anterioare ale *linux360*.

Țin să vă reamintesc doar că, cu ajutorul ROSLIMS, se pot face imagini complete sau incrementale pe CD-uri, DVD-uri sau casete DAT a hard-disk-urilor din sistemele Dumneavoastră.

De asemenea, prin intermediul lui F-Prot Antivirus (se instalează de pe Internet cu ajutorul utilitarului dedicat al ROSLIMS), distribuția poate asigura servicii de devirusare pentru toate tipurile de partiții cunoscute, *nix sau Windows.

Şi că tot veni vorba de partiții, merită menționat utilitarul QTParted cu ajutorul căruia puteți crea, modifica sau şterge partiții în mod grafic fără pierderea datelor.

NTFS

Anunțat cu surle şi trâmbițe de către producătorii săi, am încercat să folosesc Captive NTFS (inclus şi în ROSLIMS) pentru a salva câteva screenshot-uri direct pe partițiile NTFS folosite de un Windows XP. După lansarea acestuia, scanarea pentru găsirea fişierelor native Windows ce manipulează acest tip de partiții a durat în jur de 5 minute. Surpriza a venit însă la sfârşitul procesului, când serverul X a „crăpat" şi a repornit în 256 de culori, fără a se fi activat suportul de `NTFS`. Aşadar, vă recomand precauție în folosirea acestui utilitar şi evitarea pe cât posibil a acestui tip (ciudat şi foarte instabil) de partiții specific Microsoft. Dacă aveți nevoie să împărțiți date între Windows şi Linux, rămâneți la „bătrânescul" `FAT32`.

Aşadar şi prin urmare...

ROSLIMS face mai multe şi mai bine decât Knoppix. Şi, mai ales, le face în limba română. Distribuția testată este o soluție excelentă pentru PC-urile de acasă sau dintr-o firmă mică, software-ul prezent în ea îndeplinind cerințele în proporție de 99% pentru astfel de medii. Interfața localizată o face foarte prietenoasă pentru utilizatorul român şi vă va ghida prin meniuri şi programe indiferent de nivelul dvs. de experiență în ale PC-urilor.

Nu în ultimul rând, autorul vă aşteaptă cu întrebări, comentarii şi sugestii pe secțiunea dedicată ROSLIMS de pe forumul revistei.

linux360 recomandă.

Sistemul de test:
Procesor AMD Duron 1,1GHz, placă de bază ECS K7S5A, 512MB RAM, hard-disk Seagate Barracuda 120GB, DVD-ROM Toshiba SD-M1212, CD-RW Samsung SW-2525, placă grafică GeForce2 MX, placă de sunet SB Live! 7.1, placă PC-TV Wayjet 951TF-BK, placă de rețea Allied Telesyn AT-2501TX, imprimantă HP Deskjet 640C.

Resurse:

- http://www.umftgm.ro/roslims/
- ftp://ftp.linux360.ro/distributii/roslims

Autor:
ovidiu.lixandru@linux360.ro

Onebase 2004 Linux - simplitatea supremă

Alex Bucur

Onebase este o distribuție recentă - prima versiune a apărut în iulie 2003 şi Onebase 2004 a fost lansat la începutul lui ianuarie 2004 aducând îmbunătățiri majore față de versiunile precedente. A început ca o distribuție bazată pe surse dar a acceptat şi pachetele binare devenind astfel un hibrid. Nu este bazată pe nici o altă distribuție majoră sau sistem de management al pachetelor, făcând lucrurile în propriul ei stil. Este încă la începuturi dar se dovedeşte a fi o distribuție interesantă care merită urmarită în viitor.

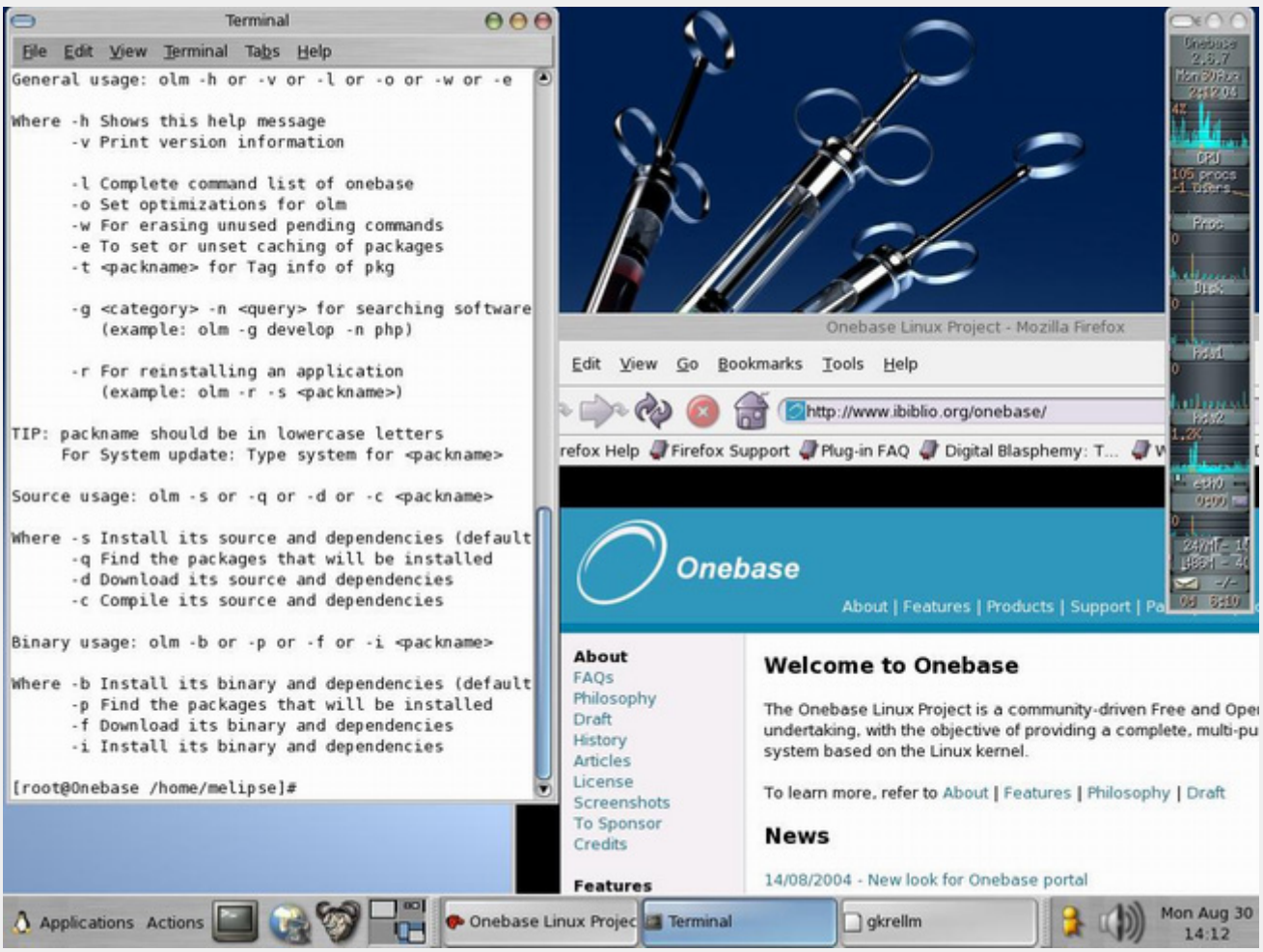
Prezentarea Olm

Înainte să începem cu instalarea, vreau să vă descriu pe scurt Olm - Onebase Linux Management. Olm şi comenzile `ol` constituie inima Onebase. Împreună, ele încearcă să fie un sistem integrat care să se ocupe de aspectele administrării sistemului, instalării inițiale şi instalării pachetelor. Acestea sunt utilitare de consolă (un GUI fiind în dezvoltare), încurajator fiind faptul că singura comandă care trebuie învațată este `olm`.

Olm poate fi utilizat pentru a instala programe atât în format sursă cât şi binar, totul depinzând de alegerea utilizatorului. Deci, `olm -s numelepachetului` o să instaleze programul din surse cu dependințele aferente, iar `olm -b` va face acelaşi lucru numai că va folosi pachete binare.

Gentoo are portage, Source Mage are grimoire - Onebase foloseşte Application Gallery. Este o colecție de informații necesare instalării pachetelor în formatul `numepachet.olm`. O să le numesc `.olm`s ca să nu fie confundate cu comanda `olm`. Deci fiecare `.olm` este echivalentul unei vrăji in Source Mage sau a `ebuild`-ului din Gentoo. Aceşti `.olm`s sunt grupați în categorii. Pentru a actualiza galeria este necesară comanda `ol-apps -u`.

Când lansezi o „vrajă” în Source Mage, dacă sunt dependințe adiționale te întreabă dacă vrei să le instalezi. Ei bine acest



lucru nu există în Onebase, dependențele fiind instalate automat.

Dacă mă gândesc bine, `olm` se ocupă cu managementul pachetelor iar comenzile `ol` (`ol-connect`, `ol-manage`, `ol-ki`, `ol-media`, `ol-desk`) se ocupă de legătura la Internet, de administrarea sistemului, reconfigurarea kernelului, alegerea driverelor de sunet (ALSA sau OSS) şi configurarea desktopului.

Instalarea

Instalarea am făcut-o pe un sistem Athlon XP 1600+, 256MB RAM, HDD 10GB, Radeon 7500LE şi CD-RW Sony.

Instalarea "default" este simplă, bootând de pe CD ajungi să alegi între o instalare nouă, să continui instalarea de unde ai rămas sau în mod "rescue". Alegând instalare, ajungem la `cfdisk`, pentru partiționare fiind disponibile ca sisteme de fişiere `ext3`, `reiserfs` (recomandat) şi `XFS`. La boot-managere se poate alege între `LILO` şi `GRUB`, primul fiind recomandat având şi `bootsplash`. Următoare sunt setările folosite pentru compilare: tipul procesorului şi nivelele de optimizare. După aceea ajungi într-o consolă în care ți se indică patru comenzi: `ol-connect` (pentru conectarea la Internet), `olm -s/-b olm` (updatarea galeriei), `olm -s/-b basepacks` (instalarea pachetelor minim necesare), `olm -s/-b linux` (instalarea unui kernel). După ce s-a terminat totul, introduceți `passwd` pentru a schimba parola şi rebootați calculatorul.

După reboot se observă faptul că Onebase ascunde mesajele de bootare sub un *splash screen*. Dacă apăsăm `F2`, vedem o rutină de auto-configurare bazată pe Knoppix. O singură problema este: această rutină începe la fiecare pornire de sistem, dacă vreți să modificați `fstab`-ul iar acesta să rămână aşa cum l-ați modificat, este necesară comanda `ol-media -f` care anulează `auto fstab`-ul, iar pentru adăugarea de noi servicii `init` este necesară comanda `ol-init`.

Instalând GNOME, am observat că implicit Onebase instalează X.org în loc de XFree86, acest lucru putând fi evitat dacă înainte de instalarea GNOME instalam în prealabil XFree86.

Application Gallery nu este prea mare, fiind nevoie de multă muncă pentru a ajunge la măcar un sfert din `portage`-ul Gentoo-ului. Acest dezavantaj se datorează faptului că este încă o comunitate destul de mică, fiind necesară implicarea din partea mai multor persoane pentru crearea de pachete; lucru care nu este greu, fiind puse la dispoziția utilizatorului şi ghiduri pentru acest lucru.

Kernelul instalat predefinit este imens, fiind necesară comanda `ol-ki` pentru reducerea dimensiunilor, dar nu creează un nou kernel cu alt nume şi cu propria intrare în `lilo.conf` ci pur şi simplu rescrie vechiul kernel.

Concluzii

Majoritatea conceptelor folosite în Onebase se găsesc şi în alte distribuții. Sorcerer îți va compila orice număr de programe în paralel, Rock Linux va lucra cu pachete ori sursă ori binare într-un mod asemănător, şi inevitabil o să vină careva care să spună că oricum Gentoo este cel mai bun. Oricum sisteme ca Sorcerer şi Rock Linux pot fi intimidante pentru persoane care nu sunt administratori de sistem profesionişti. Pe de altă parte Onebase are potențialul de a aduce utilizatorilor "normali" puterea distribuțiilor "source based" (cu posibilitatea de a instala şi pachete binare) fără a face compromisuri.

Resurse:

- http://www.onebaselinux.org
- http://www.osnews.com
- http://ftp.linux360.ro/distributii/onebase

Autor:

alex.bucur@linux360.ro

Instalare Linux prin FTP

Andrei - Lorentz Ciubotică

Linux-ul este probabil unul dintre cele mai flexibile sisteme de operare, atunci când este vorba de instalare şi nu numai. În acest articol vă voi explica, într-o manieră cât mai accesibilă, paşii ce ar trebui urmați pentru a finaliza instalarea unui sistem Linux folosind reteaua (intranet şi/sau Internet), mai exact folosind protocolul **F**ile **T**ransfer **P**rotocol. De ce este necesară aceasta? Gândiți-vă numai la cei care nu au o unitate CD-ROM ...

Trebuie să ştiti că nu toate distribuțiile Linux, chiar dacă sunt unele dintre cele mai populare, suportă modalitatea de instalare prin intermediul protocolului FTP. În consecință, m-am oprit doar la doua distribuții (RedHat şi SuSE) care suportă (şi) această facilitate.

Considerații teoretice

Pe calculatorul unde urmează a fi instalat sistemul de operare, trebuie să ruleze un Linux, minimal ce-i drept, dar trebuie să existe. Linux-ul nostru minimal, trebuie de asemeni să ştie de comunicarea prin rețea, să fie client pentru un server de FTP accesibil Linux-ului nostru, iar daca s-au îndeplinit toate aceste condiții, să lanseze în execuție utilitarul ce se ocupă cu instalarea. Aceştia ar fi în mare paşii.

În continuare, ne propunem să particularizăm pentru fiecare dintre cele două distribuții amintite mai sus, modalitatea de a instala ...

Să începem cu distribuția SuSE Linux

Avem două posibilități:

- să folosim primul CD din componența ditribuției şi să bootăm de pe acesta
- să folosim floppy disk-urile create în prealabil pe o altă maşină.

Ambele variante sunt bune, dar în funcție de situație se va opta fie pentru prima, fie pentru a doua soluție. Exemplu concret:

Am avut primul CD din distribuție (era un SuSE 9.1) zgâriat, astfel că în anumite sectoare nu se puteau citi datele. Practic, după ce boot-a CD-ul, instalarea se bloca chiar înainte de selecția pachetelor. Nu ştiam exact ce fişiere sunt afectate, aşa că am încercat pe o altă stație să copiez conținutul CD-ului. La copiere am văzut exact care sunt fişierele corupte, (erau destul de multe, majoriatatea făcând parte din componența utilitarului Yast) dar unul singur era cel critic (în cazul meu), şi anume directorul `media.1` din rădăcină. Nu putea fi accesat sub nici o formă. După ce m-am uitat pe celelalte CD-uri, mi-am dat seama care era structura directorului `media.1` şi cele 2 fişiere conținute de acesta: `media` şi `products`. Le-am editat, salvat şi le-am pus pe un server FTP.

Dacă nu aş fi avut posibilitatea de a boot-a de pe CD, as fi creat disketele de boot pe o alta statie, folosind utilitarul `rawrite` (`/dosutils/rawrite` sau `/dosutils/rawwritewin`, depinde de context) şi imaginile floppy din `/boot`. Am rămas oarecum surprins să constat că numai kernel-ul ocupă 3 diskete, pe lângă cele 4 diskete cu modulele aditionale. Dintr-un calcul simplu, rezultă că avem nevoie de cel mult 7 diskete, dar în practică, în funcție de sistemul fiecăruia, acest număr se poate reduce chiar la 4. Pe sistemul meu, pentru instalarea prin FTP, după încărcarea kernel-ului, am avut nevoie să încarc floppydisk-ul cu driverele pentru interfața de rețea (`/boot/modules3`). Aşa am ajuns la exact 4 diskete.

Pentru partea de server

Acum să pregătim maşina de pe care se va instala Linux-ul nostru. Pentru început trebuie să copiem toate cele 5 CD-uri într-un director, să-i spunem `install`. Pentru fiecare CD în parte, voi crea câte un subfolder, numit: `cd1`, `cd2`, ..., `cd5`. Server-ul FTP ales a fost: `vsftpd`. Este suficient pentru ce aveam noi nevoie, iar directorul `HOME` a fost chiar `/calea/catre/install`, creat în prealabil. Altfel spus: `#usermod ftp -d /home/install`, cu drepturile corespunzatoare.

Utilitarul de instalare va face o cautare recursiva dupa directoarele `media.[1-5]`, dupa care va porni instalarea.

RedHat şi Fedora Core

Pe acelaşi principiu ne vom ghida şi pentru distribuția RedHat/Fedora Core, dar cu câteva modificări: vom scrie pe un mediu optic reinscriptibil imaginea: `<CD1>/images/boot.iso`, în locul disketelor de boot. Modalitatea este asemanatoare ca şi în primul caz: boot-area de pe acest mediu, selecția modalității de instalare, specificarea adresei server-ului FTP de pe care se face instalarea.

În practică, dacă server-ul FTP este localizat în intranet, ar fi de preferat folosirea sa decât clasicile medii optice, evident din rațiune de viteză. Alt motiv ar fi următorul: dacă avem nevoie de o instalare Linux minimală, avem legatură la Internet destul de bună atunci nu se justifică arderea CD-urilor aferente, ci folosirea soluției prezentate în acest articol.

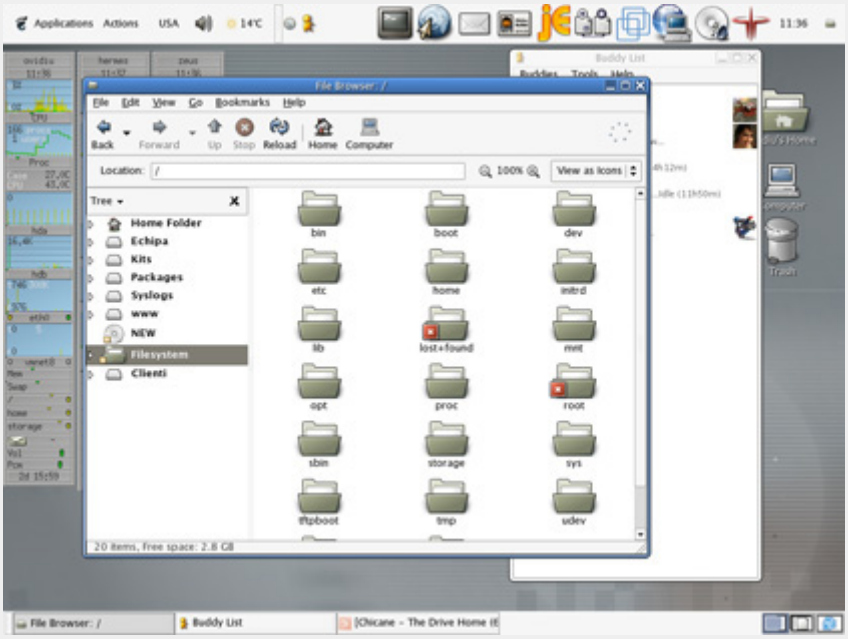
Autor:

andrei.ciubotica@linux360.ro

O viziune industrială

Ovidiu Lixandru

Recunosc, sunt un împătimit al personalizării interfețelor. Dacă pe platformele Windows interfața Aqua este favorita mea (ştiți voi, vrabia mălai visează), pe Linux este Industrial.



Cine-i Căpitanul Industrial şi de ce-mi accesează hard-disk-ul?

Tăticul look-ului Industrial este Ximian. Vă mai aduceți aminte de Ximian Desktop, „chestia" aceea care se mai găseşte acum doar pentru SuSE? De acolo a pornit totul.

Câțiva ani mai târziu după apariția XD, Red Hat au bătut cu pumnul în masă şi au strigat „Vrem o interfață unificată pentru Red Hat Linux 8". Designerii au sărit speriați şi au nominalizat două candidate: Industrial şi BlueCurve. Criteriile selecției şi cantitatea de sânge cursă la luptă n-au răsuflat până la noi. Cert e că le-a plăcut mai mult 3D-ul... E totuşi păcat fiindcă, dacă interfața alb-albastră ar fi fost aleasă în detrimentul BlueCurve, ea ar fi căpătat şi o temă pentru Qt şi kwin, cu alte cuvinte – o temă KDE. Dar aşa...

În zilele noastre, Industrial este un theme-engine pentru aplicațiile bazate pe Gimp Toolkit, în toate versiunile sale. Un theme-engine este un motor de randare ce se ocupă de desenarea şi afişarea diverselor elemente ale interfeței, în cazul Industrial atât de randarea celor GTK, cât şi GTK2. Asta înseamnă că orice aplicație bazată pe unul din aceste toolkit-uri va beneficia de facelift, de la bătrânul XMMS până la cel mai nou Bluefish.

Parolă „nimic"

Industrial vă preia încă de la primul contact cu interfața grafică - managerul de login, în cazul nostru gdm. Albastră, simplă, plăcută ochiului şi asigurându-te că o să ai ce vedea şi după login.

Ferestre nouăş...

Industrial include de asemenea o temă pentru managerul de ferestre predefinit GNOME (Metacity), una pentru managerul de fişiere Nautilus, ca şi un set de iconițe complet pentru sistem, extinzându-l pe cel original „hicolor". Toate sunt extrem de îngrijit lucrate şi îți dau un sentiment puternic de *nix.

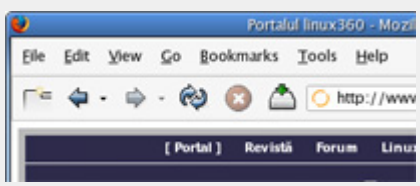
La muzică, înainte

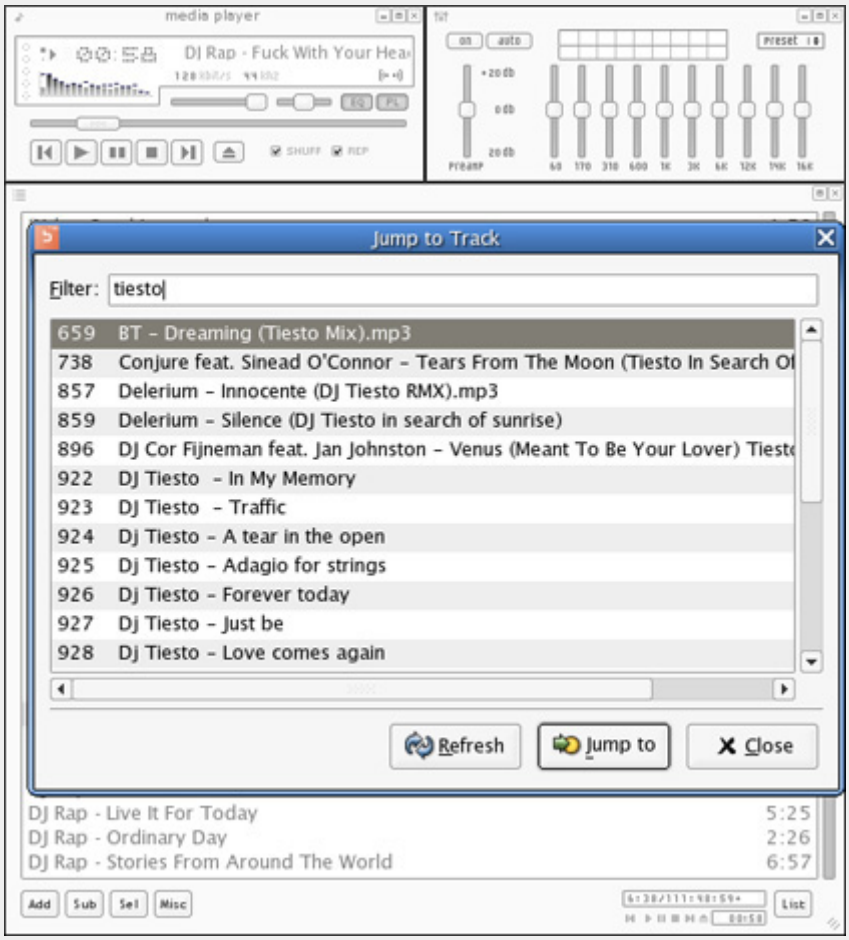
Cum aplicația de bază într-un sistem desktop (player-ul audio, să fim înțeleşi) nu se putea să ducă dorul unui skin asortat, Industrial livrează unul în format standard Winamp - *.wsz (un zip chior cu tema adică). Acesta se poate mufa fără probleme atât pe XMMS cât şi pe noul Beep Media Player.

Un browser, doi browseri

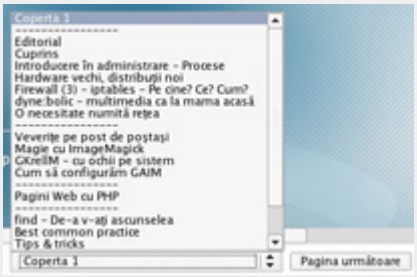
Browserele native GNOME (Epiphany, Galeon) beneficiază în mod predefinit de look-ul Industrial. Dar poate folosiți un Firefox. Caz în care va trebui un pic de tweaking.

Browser-ul de la Mozilla este scris pentru GTK+ deci, folosind skin-ul predefinit, nu veți avea nici o problemă cu look-ul controalelor. Totuşi, icoanele temei stock parcă nu îşi au locul în Industrial. Nici o problemă! Novell au lansat de curând o temă pentru Firefox denumită chiar „Industrial". Aceasta va aduce în browser grafica temei de sistem, ducând la o integrare completă.





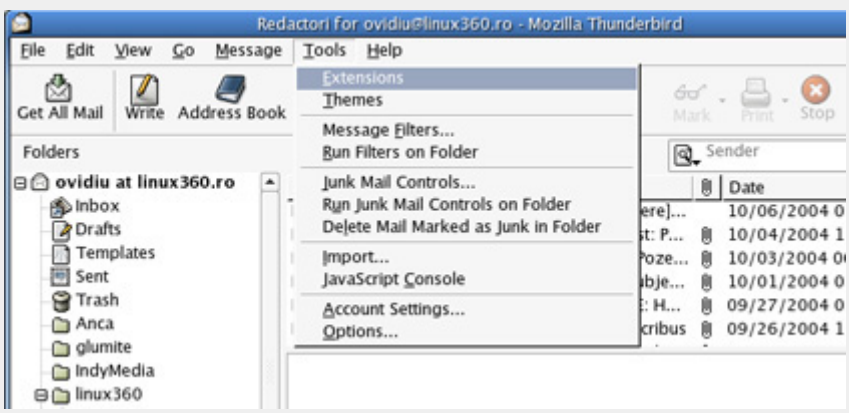
Perfecționiștii mai au totuşi un pas de făcut - controalele din form-uri. Aceste controale nu sunt modificate indiferent de skin, lăsând în seama browser-ului randarea lor. Iar browser-ul randează, pe o platformă *nix, cu widget-urile din Windows. Chestiune destul de bizară. Tigert a lansat însă de curând un „patch” (mai mult un mod decât un patch) numit firefox-forms care, prin înlocuirea câtorva fişiere ale lui Firefox (aveți grijă să vă faceți backup, poate veți dori să reveniți la look-ul original cu arome de Redmont - da, siguuur), îl determină să randeze form-urile folosind look-ul Industrial.



Poştaşul electronic

Aplicăm aceeaşi lemă ca în cazul browserelor: dacă e aplicație GTK nativă, va fi şi „industrializată”. Evolution e deci pregătit.

Thunderbird are aceeaşi bubă ca şi Firefox. Preia controalele, dar iconițele din tema predefinită a aplicației sunt ca nuca-n perete. În cazul clientului de mail nu mai există însă o temă completă, ci doar una „la trei sferturi”. GNOME-Tb, o temă mai veche pentru Thunderbird, va aduce în scenă iconițele hi-color, dar, dintr-un motiv cel puțin dubios, nu preia şi look-ul meniurilor, păstrându-l însă pe cel al butoanelor şi al restului de elemente ale interfeței. **„Când e gata Vision?”**



Vă întrebați de ce-am adus în discuție Vision aici? Apăi cum vă întrebați asta după ce-ați citit titlul articolului? Şi nu în ultimul rând, distribuția linux360 veşnic-în-pregătire are ca temă predefinită tocmai subiectul articolului - Industrial. Cu toate temele, skin-urile şi mod-urile sale. Deci, dacă mai aveți răbdare încă o țără, veți putea încerca Industrial direct pe o distribuție completă.

Vreau acum!

Mă bucur să aud că v-am stârnit interesul. Veți găsi totul pe server-ul FTP linux360, gata de download şi instalare. Aveți nevoie doar de un GNOME (2.6 sau mai nou recomandat) şi aplicațiile pomenite mai devreme. Artwork-ul este disponibil în formatele src.rpm şi rpm, deci aveți grijă să aveți şi ceva cu care să-l desfaceți/recompilați sau să-l instalați, după caz. Personalizare plăcută şi hai lansare Vision!

P.S. V-am spus că-mi place să personalizez interfețe?

Resurse:

- http://ftp.linux360.ro/distributii/vision-1.0/industrial

Autor:

ovidiu.lixandru@linux360.ro

Idesk - o iconiță pentru fiecare

Anca-Mihaela Holban

O iconiță, două iconițe... Dar ce faci când n-ai ce număra?

Din cauza frustrărilor provocate în trecut de performanțele inexistente ale sistemului pe care-l aveam, folosirea unui desktop ca GNOME sau KDE fiind imposibilă, am fost nevoită să folosesc window managere care nu sunt atât de înfometate de resurse. Acum, deşi am alt PC mult mai bun, folosesc fluxbox. Şi vreau iconițe.

Mooove your body giiirrl

Vreau iconițe şi le voi avea folosind Idesk. Idesk permite adăugarea iconițelor la fereastra root a unei sesiuni X. Iconițele pot avea orice dimensiuni, putând fi plasate oriunde doriți, pot deveni transparente la trecerea mouse-ului peste ele etc.. Pot fi configurate şi alte atribute ca font-ul, mărimea font-ului, umbra.

Instalare

După descărcarea sursei de pe site, aceasta trebuie dezarhivată într-un director, apoi trebuie rulată în acelaşi director comanda make, iar apoi, ca root, make install.

```
$ make
$ su
# make install
```

make install va copia fişierul executabil în `/usr/local/bin/idesk`.

Configurare

Pentru a putea folosi Idesk trebuie să creați un fişier de configurare `.ideskrc` în directorul home. Conținutul acestui fişier va fi:

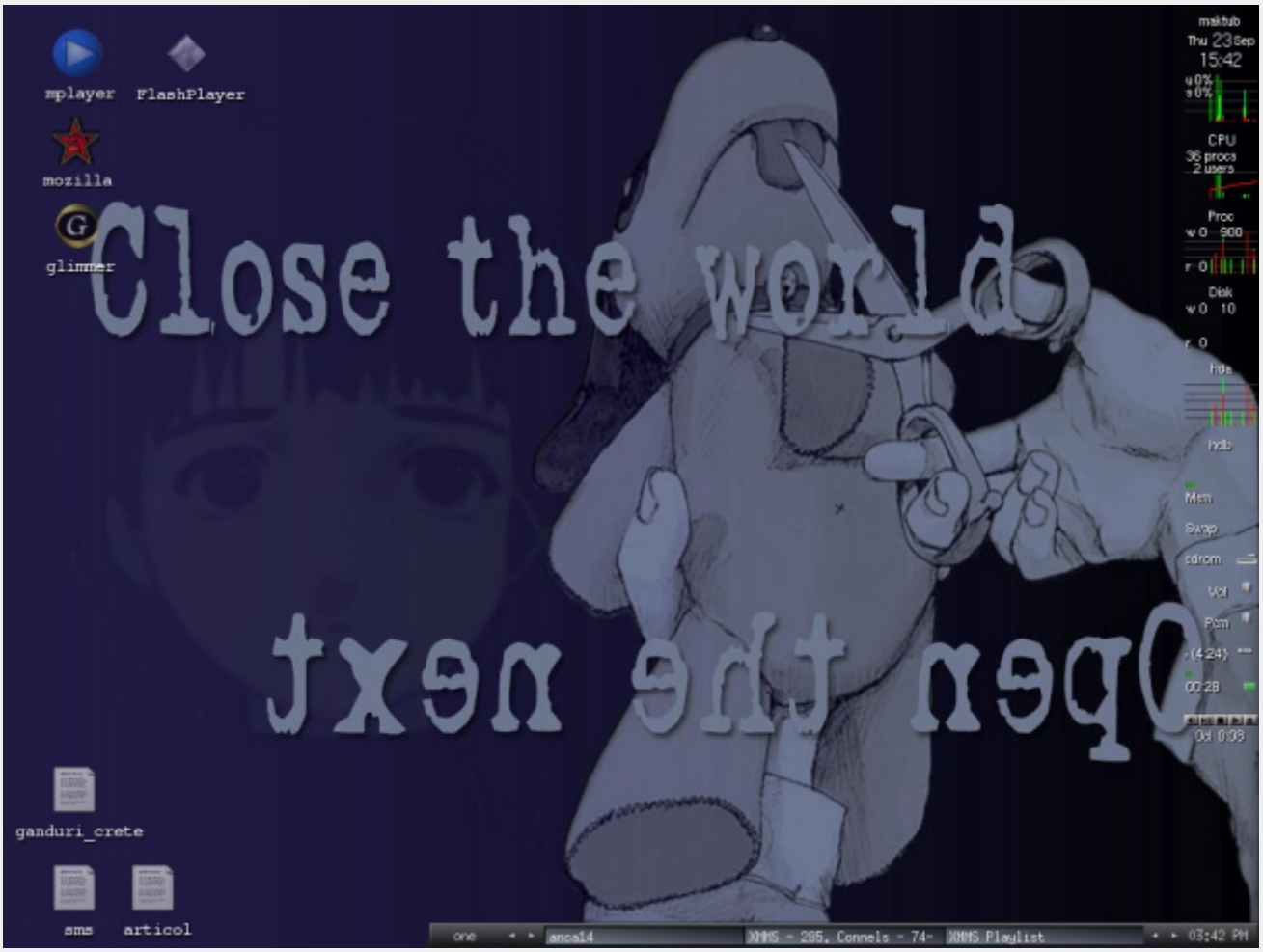
```
table Config
  FontName: tahoma
  FontSize: 8
  FontColor: #ffffff
  Locked: false
  Transparency: 150
  Shadow: true
  ShadowColor: #000000
  ShadowX: 1
  ShadowY: 2
  Bold: false
  ClickDelay: 300
  IconSnap: true
  SnapWidth: 55
  SnapHeight: 100
  SnapOrigin: BottomRight
  SnapShadow: true
  SnapShadowTrans: 200
  CaptionOnHover: false
  end

table Actions
      Lock: control right
doubleClk
  Reload: middle doubleClk
  Drag: left hold
  EndDrag: left singleClk
  Execute[0]: left doubleClk
        Execute[1]: right
doubleClk
end
```

- `FontName` - fontul textului care va servi la descrierea iconiței
- `FontSize` - mărimea fontului (duh!).
- `FontColor` - codul culorii textului.
- `Locked` - poate fi setat ca true sau false. Este util să îi dați valoarea true pentru a putea muta iconițele, iar după ce le veți fi aşezat să îi atribuiți valoarea false.
- `Transparency` - valoarea atribuită poate fi între 0 (fără transparență) şi 255 (invizibilă atunci când mouse-ul nu este pe iconiță).
- `Shadow` - la fel ca `Locked`, poate fi setat ca true sau false. Dacă este setat true, textul va avea o umbră de culoarea. `ShadowColor`, având în dreapta grosimea `ShadowX` pixeli, iar în jos `ShadowY`. Valorile implicite ale `ShadowX` şi `ShadowY` sunt 1, acestea putând fi setate şi negative.
- `Bold` - dacă îl setați ca true textul va apărea îngroşat.
- `Click Delay` - numărul de milisecunde în care două click-uri succesive sunt considerate dublu-click
- `SingleClick` - setat ca true, va fi nevoie de un singur click pe iconiță pentru a executa comanda asociată acesteia. Ca o urmare a acestei setări, mutarea iconițelor prin drag and drop nu mai este posibilă.
- `IconSnap` - setat ca true, face ca iconița să fie aranjeză în mijlocul unui dreptunghi definit de `SnapWidth` şi `SnapHeight`. `SnapWidth` şi `SnapHeight` trebuie să aibă valori mai mari sau egale decât dimensiunile iconiței.
- `SnapWidth` - Lățimea dreptunghiului.
- `SnapHeight` - Înălțimea dreptunghiului.
- `SnapOrigin` arată de unde începe dreptunghiul, putând avea valorile `TopLeft`, `TopRight`, `BottomLeft`, `BottomRight`.
- `SnapShadowTrans` - poate lua valori de la 0 (opac) la 255 (invizibil).
- `CaptionOnHover` - setat ca true face ca descrierea iconiței să fie vizibilă numai când mouse-ul se află peste iconiță.

Acțiunile care se pot face asupra unei iconițe:

- `Lock`: iconița nu mai poate fi mutată.
- `Reload`: reîncarcă opțiunile de configurare.
- `Drag`: iconița respectivă va urma cursorul până la EndDrag.



- `Execute[i]`: Execută comanda asociată iconiței. `Execute[0]` execută `Command[0]` din fişierul de configurare al iconiței, `Execute[1]` execută `Command[1]` etc..

Opțiunile pentru fiecare comandă pot fi:

`<shift> <control> <alt> [Button] [ClickType]`
unde:

- `[Button]` = `left`, `middle`, sau `right`
- `[ClickType]` = `singleClk`, `doubleClk`, `tripleClk`, sau `hold`

Să curgă iconițele

Pentru a adăuga iconițe trebuie să creați directorul .idesktop în directorul home, iar în acel director să creați fişiere de configurare cu extensia .lnk pentru fiecare iconița în parte. În directorul ~/.idesktop am creat fişierul articol.lnk:

```
table icon
  Caption: articol
  Icon: ~/icons/txt.png
  X: 105
  Y: 700
      Command[0]: abiword
~/1360/idesk/art.txt
     Command[1]: aterm -e
~/1360/idesk/art.txt
end
```

- `Caption` - descrierea iconiței.
- `Icon` - calea imaginii folosită ca iconiță (Idesk suportă numai imagini cu extensia png sau svg.
- `X, Y` - coordonatele unde va fi poziționată iconița.
- `Command[0]`: aplicația executată când asupra iconiței se face acțiunea `Execute[0]` (din `~/.ideskrc`).

Pentru celelalte iconițe va trebui să creați alte fişiere cu extensia lnk după modelul celui de mai sus.

Eu am ce număra. Sper că şi voi.

Resurse:

- http://idesk.timmfin.net

Autor:

anca.holban@linux360.ro

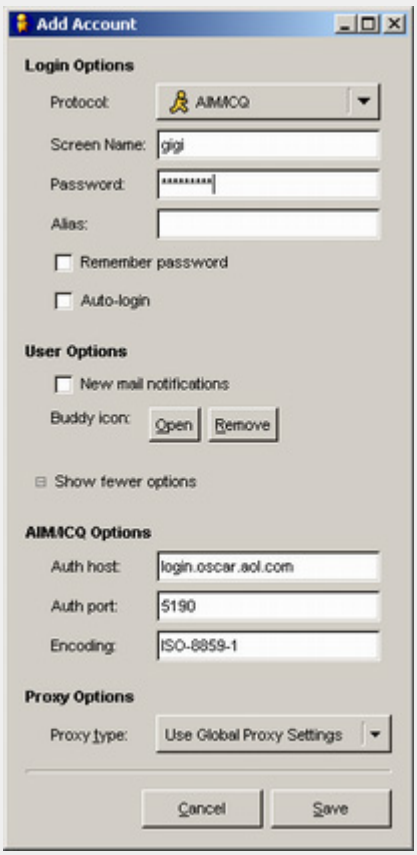
GAIM - Configurare (2)

Dan Marcu şi Radu - Eosif Mihăilescu

Aşa cum spuneam în numărul precendent, iată-ne ajunşi acum în postura de a configura clientul multifuncțional de mesagerie instantă, GAIM, pentru funcționarea curentă (în episodul trecut v-am arătat cum să-i setați opțiunile şi preferințele generale).

După cum poate ați observat în decursul utilizării, GAIM are o structură modulară -- structură a cărei unitate morfologică şi funcțională de bază este "plugin"-ul. Avem astfel plugin-uri de uz general şi plugin-uri care deservesc (asigură suportul pentru şi transcomunicația cu) câte un protocol.

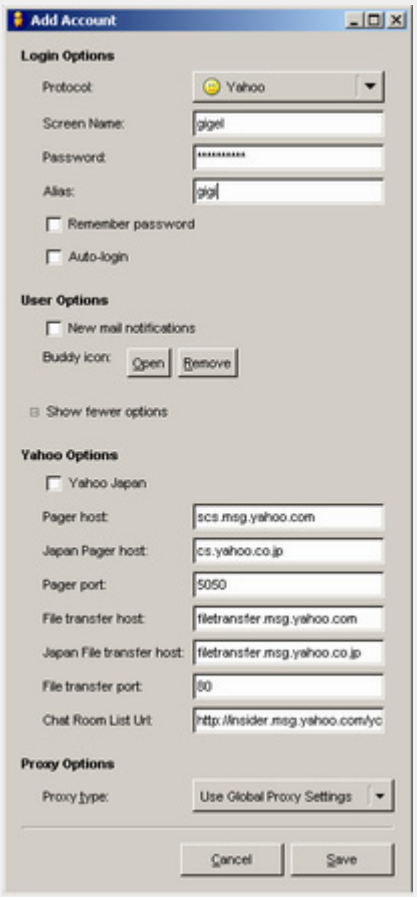
Avem un plugin pentru protocolul ICQ/AIM (sunt aproape identice şi de aceea sunt tratate în majoritatea cazurilor împreună), unul pentru protocolul MSN şi încă unul pentru Yahoo!.



De asemenea, mai există plugin-uri pentru Jabber cât şi pentru protocoale care nu sunt considerate în mod normal "de mesagerie instantă" cum ar fi IRC, Napster etc.

După cum se poate vedea în figurile alăturate, configurarea pentru ICQ/AIM şi MSN seamănă izbitor cu cea din clientul nativ (de Windows) cu doar câteva excepții cum ar fi posibilitatea de a specifica numele (sau adresa IP) a server-ului pentru protocolul respectiv.

În plus față de setările cu care eram familiari, mai avem posibilitatea de a ne alege setul de caractere implicit la ICQ/AIM precum şi posibilitatea de a selecta încapsularea protocolului MSN în HTTP (facilitate utilă dacă ne aflăm în spatele unui firewall ce nu permite conexiuni pe portul 1869).



Poate cea mai înspăimântătoare dintre toate, pare a fi cutia de la plugin-ul de Yahoo! care ne prezintă, la prima vedere, un număr foarte mare de opțiuni față de cele prezente în clientul nativ (după unii critici: "nici una").

Nu avem nici un motiv să ne speriem, ni se dă doar posibilitatea de a specifica manual numele server-ului principal şi al celui pentru transfer de fişiere de la Yahoo! precum si porturile pe care aceste servicii rulează.

Autori:

dan.marcu@linux360.ro
radu.mihailescu@linux360.ro

Sisteme de control al reviziilor

Radu - Eosif Mihăilescu

Poate mulți dintre dumneavoastră au auzit, măcar şi o singură dată, pe pagina de web a unui program oarecare, sintagma: "*aveți aici versiunea stabilă şi aici ultima versiune din* `CVS`". Foarte bine şi frumos, dar ce e `CVS`?

Urmând un raționament simplu bazat pe colectarea şi compararea de exemple, poate ați dedus că "`CVS`" are ceva de a face cu menținerea şi întreținerea surselor programului în timpul dezvoltării sale de către programatori. Poate v-ați dat seama că este ca un fel de autoritate centrală care supervizează procesul de modificare al surselor programului.

În fine, poate ați întâlnit termenii de *check in* şi *check out* citind liste de discuții şi poate v-ați dat seama că se refereau la cele două tranzacții primordiale pe care le suportă un depozit `CVS`.

Totuşi, ce e ăla `CVS`? Ei bine, seria de articole deschisă de cel pe care îl citiți acum îşi propune să vă răspundă, pe larg, la această întrebare.

Vom începe cu o descriere generală a tuturor conceptelor ce stau la baza sistemelor de control al reviziilor şi vom prezenta câteva unelte (acum de domeniu istoric) ce au fost folosite inițial în acest scop. În numărul următor va fi abordat `CVS`, standardul de facto în ceea ce priveşte sistemele de care vorbim urmând ca în ultimul episod să-l tratăm pe `SVN`, viitorul `CVS`-ului aşa cum pare astăzi.

În funcție de părerea dumneavoastră, a publicului cititor, seria va mai putea fi extinsă cu un al patrulea episod care să acopere alte sisteme de acest gen folosite astăzi cum ar fi `BitKeeper` ş.a.

Concepte de bază

Conceptul pe care se bazează orice sistem de control al reviziilor este *depozitul*. Acesta este un loc în care sunt ținute, stocate, *depozitate* toate obiectele aflate sub controlul sistemului în cauză.

Aceste obiecte pot fi de mai multe (generalizând: de orice) feluri, dar pentru a simplifica explicația, vom presupune că sunt fişiere şi directoare.

"*Hmm, păi acesta este un server de fişiere*" veți spune şi aveți dreptate pentru că un depozit al unui sistem de control al reviziilor este un server de fişiere - dar unul aparte, unul cu o proprietate ce îl face special: are capacitatea de a memora toate schimbările care se aduc spațiului de nomenclatură conținut şi chiar fiecărui obiect în parte.

Atunci când accesăm un server de fişiere obişnuit, vedem fiecare fişier aşa cum se află în prezent – în cazul unui sistem de control al reviziilor însă, am putea face o interogare de genul "*îl vreau pe* `/README.txt` *aşa cum arăta ieri*" şi sistemul ne va pune la dispoziție acea versiune (sau revizie) a fişierului în cauză.

Aceasta este doar una din proprietățile ce disting un server de fişiere de un sistem de control al reviziilor, următoarele sunt şi mai interesante.

Modele de revizii

Scopul primordial al unui sistem de control al reviziilor este să permită editarea simultană şi partajarea informațiilor.

Toate sistemele de acest gen au de rezolvat aceeaşi problemă fundamentală: "*cum va reuşi sistemul să permită utilizatorilor să partajeze informații dar fără a se încurca reciproc şi fără a-şi distruge reciproc schimbările efectuate?*"

Problema partajării fişierelor

Să presupunem că avem un server de fişiere şi doi utilizatori. Să mai presupunem că cei doi au de lucrat la acelaşi fişier.

Cei doi accesează simultan depozitul şi îşi copiază local fişierul în cauză urmând a se apuca de lucru fiecare în parte. După ceva timp, primul utilizator îşi termină de efectuat modificările necesare şi salvează la loc în depozit fişierul cu pricina. După încă ceva timp, acelaşi lucru se întâmplă şi în cazul celui de al doilea utilizator, dar... dezastru: în momentul când şi acesta îşi salvează munca în depozit, schimbările făcute de primul utilizator s-au pierdut deoarece al doilea utilizator avea o copie a fişierului inițial – fişier în care nu apar acele modificări.

Soluția blochează - modifică - eliberează

Spre rezolvarea problemei expuse anterior, unele sisteme de control a reviziilor folosesc un model de acces bazat pe blocaje. Astfel, atunci când primul utilizator dorea să modifice un anume fişier, acesta ar fi instruit sistemul să blocheze fişierul în cauză la scriere – cel de al doilea utilizator fiind astfel în imposibilitatea de a salva peste fişierul blocat.

Această soluție, deşi pare a rezova problema, induce alte probleme care au făcut să nu fie adoptată pe scară largă, cum ar fi:

- blocarea poate produce probleme administrative. Gândiți-vă ce s-ar întâmpla daca primul utilizator ar uita că a blocat acel fişier şi deci nu ar mai elibera blocajul niciodată
- blocarea poate produce uneori "cozi de aşteptare" inutile. Presupunând că primul utilizator avea treabă la începutul

fişierului, iar cel de al doilea, la coada sa, nu este nici un motiv obiectiv pentru care cele două schimbări nu ar putea să fie efectuate simultan (presupunând că ar fi "*lipite*" ulterior una de alta).

Soluția copiază - modifică - combină

Multe sisteme de control a reviziilor contemporane folosesc sistemul amintit ca o alternativă la blocare.

Cum funcționează aceasta? Foarte simplu: revenind la exemplul inițial cu cei doi utilizatori, să presupunem din nou că cei doi accesează sistemul şi îşi copiază local fişierul în cauză.

În contextul soluției discutate, această copie locală se numeşte *copie de lucru* a fişierului. Operațiunea prin care această copie locală ia naştere se numeşte "*check out*".

Din nou, primul utilizator modifică fişierul şi trimite noua versiune la sistem. Această operație se numeşte "*check in*". Sistemul compară versiunea proprie cu cea trimisă de utilizator, calculează şi stochează diferențele dintre cele două şi o marchează pe cea din urmă ca *cea mai recentă* – termenul englezesc este "*HEAD revision*".

Cel de al doilea utilizator termină şi el de lucrat şi încearcă să facă şi el "*check in*". În acest moment, sistemul verifică (ca şi în primul caz) versiunea trimisă de utilizator şi versiunea proprie şi, **atenție**, descoperă o inadvertență: copia pe care şi-a operat modificările cel de al doilea este *expirată* (eng. "*out of date*").

De ce? Deoarece fişierul în cauză a mai "sărit" cu o versiune prin "*check in*"-ul primului utilizator.

Ce poate face al doilea utilizator? Sunt schimbările sale pierdute? Nu, deloc. E suficient ca cel de-al doilea utilizator să execute o operație de *actualizare* (eng. "*update*"), în urma căreia inadvertența de care vorbeam se va fi corectat – şi astfel, după *update*, se poate face în final şi *check in*-ul mult dorit.

Ce se întâmplă de fapt în cadrul operațiunii de "*update*"? Foarte simplu: versiunea locală este *combinată* (presupunând că acest lucru este posibil, vezi în continuare) cu cea din depozit obținându-se o versiune care conține ultimele modificări din depozit plus cele efectuate local.

Ce se întâmplă în cazul în care modificările locale se suprapun cu cele din depozit? În acest caz particular fişierul cu pricina se zice a se afla *în stare de conflict* şi rezolvarea conflictului se face manual şi cade în sarcina oamenilor.

Cam acestea ar fi conceptele de bază privind alcătuirea şi funcționarea unui sistem de gestiune a reviziilor.

Exemplu didactic – sistemul RCS

Să vedem acum un exemplu practic în persoana celui mai vechi (dacă nu cumva chiar primul) astfel de sistem, clasicul `RCS`.

`RCS` vine de la *Revision Control System* şi este un sistem foarte simplu (empiric chiar) de control al reviziilor.

Acesta foloseşte modelul de acces cu blocare discutat anterior (în literatura de specialitate mai este întâlnit şi sub sintagma *sistemul bibliotecă*) şi are un model de depozit în care fiecărui fişier aflat sub controlul sistemului îi corespunde un fişier-depozit propriu.

De aici şi primul dezavantaj al `RCS` şi anume că nu "ştie" de directoare – pentru simplul motiv că nu şi le poate reprezenta şi deci nu poate urmări schimbări în acest tip de obiecte.

Cum funcționează practic? `RCS` ține pentru fiecare fişier aflat în "evidențe" câte un fişier special având aceeaşi denumire ca fişierul urmărit şi sufixul "`,v`". Prin tradiție, aceste fişiere se numesc "fişiere `RCS`" şi rezidă într-un subdirector al directorului curent numit chiar "`./RCS`".

Operațiunea de "*check in inițial*" (cunoscută şi sub numele de "*import*") decurge, după ce ne-am creat directorul `./RCS`, prin comanda "`ci fisier`".

Operația de copiere şi blocare decurge prin "`co -l fisier`". În fine, după editare, se execută "`ci -u fisier`".

Se poate observa (şi experimenta) că, atâta timp cât un fişier este blocat, acesta nu poate fi nici măcar extras din depozit de altcineva – darămite modificat şi salvat.

Acestea fiind spuse, închei spunându-vă că data viitoare vom vorbi, aşa cum am promis, de cel mai popular astfel de sistem şi anume `CVS`.

Material bazat pe şi inspirat din "*The Subversion Book*" de Ben Collins-Sussman, Brian W. Fitzpatrick şi C. Michael Pilato.

Resurse:

- http://svnbook.red-bean.com/

Autor:

radu.mihailescu@linux360.ro

Gaming Linux - la drum!

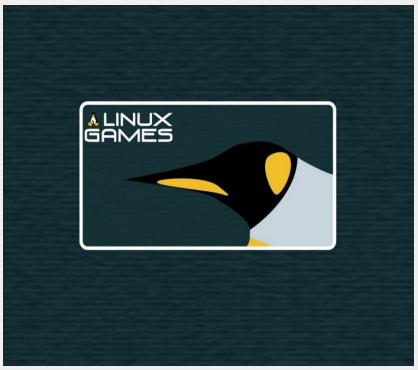
Ciprian - Alexandru Negrilă

Sincer nu ştiu cu ce să încep. Vă întrebați ce aş putea scrie despre jocurile de Linux? Destul de multe, chiar dacă este surprinzător. Aşa că voi începe pas cu pas, primul fiind afirmația că: Da, există! Şi nu numai că există dar sunt multe şi sunt şi reuşite. Nu voi începe să dau nume acum, va fi destul timp pentru asta. Tot ce voi spune acum este că am atâtea de povestit încât vom avea, în principiu, trei articole despre jocuri în fiecare numar. De ce trei? Păi pentru că sunt multe şi pentru că ne face plăcere să scriem despre lucruri distractive.

Cum stă treaba

Acum vin amănuntele. Ca o prezentare a planului acestei serii vă informez că voi încerca să cuprind 186 de jocuri în aceste articole, toate gratuite. Voi face acest lucru într-un mod cât mai plăcut şi pentru mine şi mai ales pentru voi, cititorii. Pentru aceasta vă recomand prin intermediul articolului ca atunci când auziți de un joc nou/interesant care rulează nativ pe platforma Linux sau pur şi simplu vreți să îl vedeți în revistă, să trimiteți un e-mail pe adresa redacției. Astfel veți regăsi în revistă ceea ce doriți şi nu ceea ce credem noi că e "tare".

Acum vă voi spune câte ceva despre jocurile ce vor fi cuprinse în articole şi ce trebuie să împlinească cererile voastre. În primul rând aceasta este o revistă despre Linux şi astfel jocurile trebuie să fie native acestei platforme. Aşa că nu vreau să vă prind cu cereri de articole despre Grand Theft Auto: Vice City sau Diablo II. Aici nu încercăm să vă învățăm cum să folosiți emulatoarele pentru a aduce Windows-ul pe Linux, ci încercăm să deschidem ochii comunității. Astfel, nu vor fi ignorate cereri de articole despre



It tastes better than it looks

Neverwinter Nights sau Unreal Tournament 2004. Cu toate că aceste jocuri sunt comerciale, vor fi prezentate pentru că sunt foarte, foarte cunoscute şi foarte jucate. Un alt motiv al prezentării lor este acela că, dacă unii dintre marii producători de jocuri adoptă alternativa Linux, ei sunt un bun exemplu pentru restul pieței. În definitiv, această platformă în domeniul jocurilor este o piață cu cerere foarte mare, iar puținii curajoşi care au îndraznit să scoată un titlu în acest sens sunt foarte puțini.

Să nu batem pasul pe loc şi să trecem la urmatorul punct de interes, şi anume construcția seriei. Cum spuneam şi mai devreme, această serie va cuprinde câte trei articole în fiecare număr. Aceste trei articole vor aduce informații despre jocuri tuturor categoriilor de jucători. Nu de dragul variației, ci din motive serioase. După cum bine ştiți mulți dintre voi, Linux-ul rulează pe orice arhitectură, de la i386 în sus. În mod deloc surprinzător şi jocurile merg pe aceleaşi arhitecturi. Însă nu ne putem juca din consolă jocurile care cer interfața grafică sau accelerare 3D. Din acest motiv unul dintre articole va fi destinat jocurilor de consolă. Cu dedicație pentru acei nostalgici care încă se mai joacă Rogue sau NetHack. Al doilea articol va fi destinat jocurilor care nu necesită accelerare 3D, ci doar o placă video pe care să poată rula serverul X. În sfârşit ultimul articol (care nu va lipsi sub nici o formă) se va referi la jocurile cu accelerare 3D. În această categorie vor intra cele mai noi şi mai populare jocuri.

Ce mai aşteptați?

Acestea fiind spuse, vă propun să trecem la treabă, eu la scris, voi la citit. Şi sper că ceea ce va rezulta în urma acestei serii să convingă mica noastră comunitate că: Da, se poate! şi nu numai: Da, se poate chiar bine!

Şi pentru a demonstra că cele spuse mai sus sunt adevărate, în numărul viitor voi trata trei genuri de jocuri. Un RPG cu grafica compusă din caractere ASCII, o clonă de Super Mario Bros. ăi un 3D Shooter. Vă voi lăsa pe voi să ghiciți care sunt acelea. Să auzim de bine!

Resurse:

- http://www.linuxgames.com/
- http://www.happypenguin.org/
- http://www.happypenguin.org/
- http://www.tuxgames.com/
- http://www.linux-games.com/
- http://games.linux.sk/

Autor:

ciprian.negrila@linux360.ro

Construiți cu Linux - MP3 player independent cu afişare pe LCD

Abibula Aygun

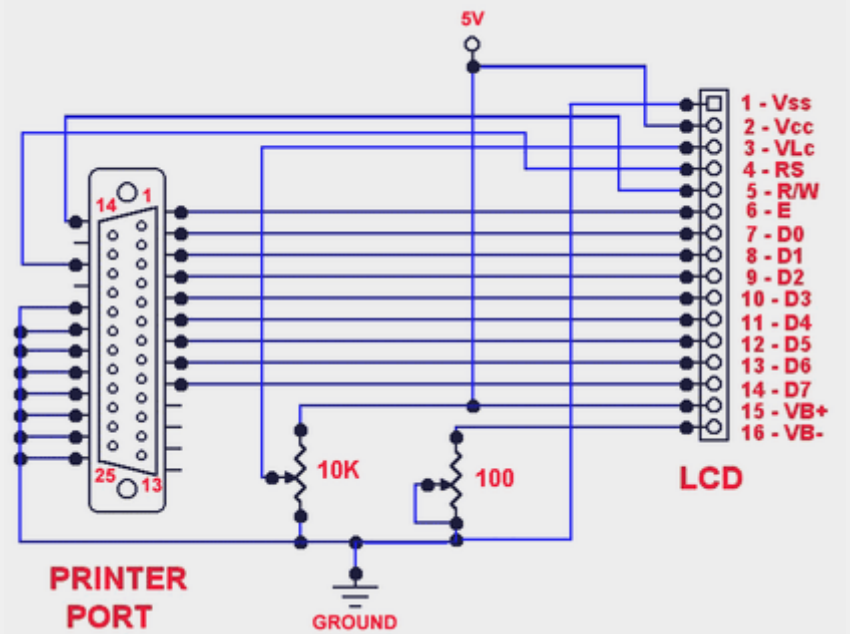
Bine v-am găsit stimați cititori la prima ediție a rubricii „Constuiți cu Linux”. Această rubrică se doreşte a fi un ghid practic pentru cei care doresc să îmbine utilul cu plăcutul şi totodată să dea o utilitate vechilor sisteme.

Acestea fiind spuse, doriți să vă construiți propriul MP3 player standalone (independent)? Dacă DA, haideți să vedem ce materiale ne trebuie pentru construcția propriu-zisă:

- o placă de bază cu cel puțin un procesor Intel DX4 rulând la 100MHz, Intel Pentium la 90MHz sau chiar mai mult, al cărei BIOS trebuie să "cunoască" bootarea de pe CD-ROM (să fie capabil a porni sistemul de calcul folosind datele de pe un CD-ROM bootabil El Torito)
- o placă de sunet Creative Labs SoundBlaster 16 ISA/PCI sau orice altă placă de sunet suportată de kernel-ul Linux
- un CD-ROM ATAPI de cel puțin `4x`

MP3 player-ul în funcțiune

- o sursă de tensiune AT/ATX. În numerele viitoare vom aborda şi construcția unei surse de tensiune capabilă să alimenteze un calculator de la bateria automobilului



Conectarea afişajului LCD

- un afişaj cu cristale lichide - LCD (Liquid Cristal Display) - de `16x2` caractere, având controler `HD44780`. Afişajul se va interfața cu sistemul prin intermediul portului paralel conform schemei din figură
- imaginea de boot de la adresa dată la "resurse"
- tastatura clasică din care se va utiliza numai secțiunea de NumPad pentru trimiterea de comenzi player-ului nostru.

Cam acestea ar fi ingredientele principale pentru construcția MP3 player-ului nostru. Nu este necesară prezența plăcii video, monitorului sau a unităților de disc de masă sau flexibil. Dupa asamblarea sistemului vom trece la scrierea CD-ului bootabil având grijă să punem în rădăcină toate fişierele MP3 pe care dorim să le ascultăm. Imaginea de boot va fi cea descărcată cu ajutorul URL-ului de la secțiunea "Resurse".

Haideți să vedem cum funcționează MP3 player-ul nostru: porniți sistemul,

Afişajul LCD în funcțiune

introduceți CD-ul făcut mai sus, computerul citeşte CD-ul (figura 1), bootează, caută fişiere MP3 şi porneşte playerul. Pe afişajul LCD putem vedea denumirea piesei şi durata.

Playerul se controlează din tastatura numerică, NumPad, ale cărei taste au următoarele funcții:

- `8` – `PLAY` (redare)
- `5` – `PAUSE` (pauză)
- `9`, `6` – `NEXT` (următoarea piesă)
- `7`, `4` – `PREV` (piesa precedentă)
- `1` – `PITCH -` (viteza piesă -)
- `3` – `PITCH +` (viteza piesă +)
- `2` – `RESET PITCH` (revenire la viteza normală)
- `+` – `VOLUME +` (volum sonor +)
- `-` – `VOLUME -` (volum sonor -)
- `.` – `JUMP TO` (sare la piesa specificată)
- `/` – `MOUNT` (montează CD)
- `*` – `UMOUNT` (demontează CD)

Viteza de lucru a CD-ROM–ului este redusă în mod automat la `2x` pentru a nu face zgomot, considerându-se că la această viteză se păstrează o rată constantă a transferului de date, suficientă pentru decodarea fluxului `MPEG1`. De asemenea, ultima variantă a imaginii de boot `mpMan`, poate să decodeze şi `MP3`-uri aflate pe hard disk cu condiția ca fişierele să se afle în directorul rădăcină, deci se poate asculta fie de pe hard disk fie de pe CD.

Hard disk-ul trebuie să fie formatat `FAT32` şi trebuie ca acesta să fie primul hard disk şi prima partiție bootabilă din sistem. În momentul bootării, sistemul caută CD-ROM-ul în `/dev/hdb iso9660`, `/dev/hdc iso9660`, `/dev/hdd iso9660`, iar dacă nu gaseşte nici un dispozitiv CD-ROM, va căuta `/dev/hda1 vfat` pentru o partiție, o va monta şi va căuta fişiere `MP3` pe aceasta.

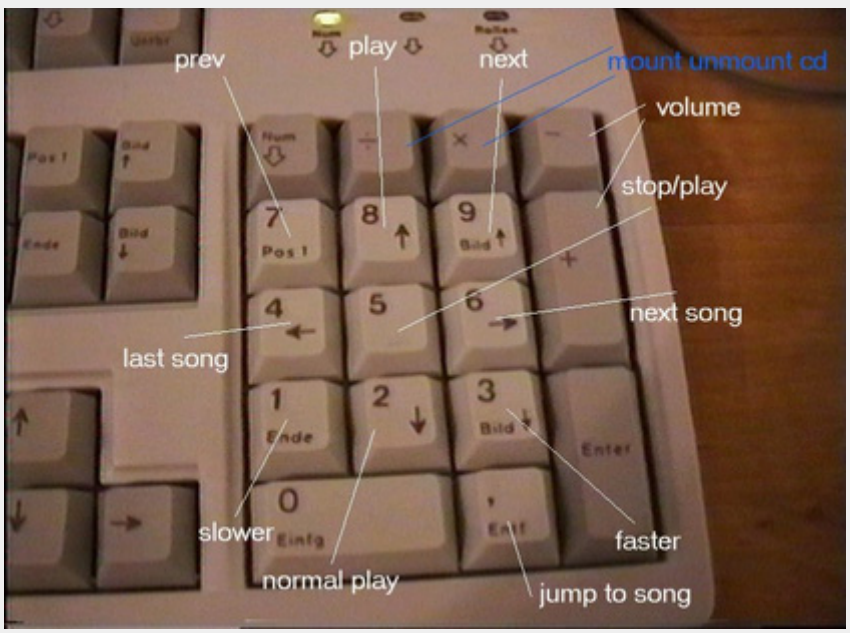
Se poate boota şi de pe hard disk dar pentru aceasta trebuie să copiați imaginea de disc (`image.dsk`) pe o dischetă. În Linux se scrie comanda `dd if=image.dsk of=/dev/fd0`, moment în care va fi copiată imaginea de disc pe floppy. Acum, transferați fişierele de pe floppy disc în partiția `C:` a hard disk-ului bootabil.

Resurse:

- http://home.t-online.de/home/mirkoroller/mpMan/image4_mar_2001.zip

Autor:

aygun.abibula@linux360.ro



Utilizarea tastaturii pentru control

HTML - Organizarea informațiilor folosind tabele

Cristian Bidea

În numărul trecut am arătat cum se "scriu" tabelele în HTML şi am menționat faptul că tabelele sunt intens folosite pentru design-ul paginilor WEB. În acest număr voi încerca să arăt câteva situații în care tabelele sunt extrem de folositoare în aranjarea informațiilor într-o pagină WEB.

Pentru aceasta am să arăt cum construim prima pagină dintr-un site. Am ales ca temă site-ul revistei linux360. Nu voi intra în argumente de design si nu voi face acest lucru din mai multe motive: în primul rand nu sunt designer, în al doilea rând nu ştiu cum să motivez acțiunile referitoare la design pe care le-am întreprins în construcția acestei pagini. Ceea ce trebuie să aveți în vedere în timpul citirii acestui articol este faptul că de cele mai multe ori "arta" se fură, nu se inventează. Nu vă speriați! Prin cele spuse anterior nu fac un îndemn la cine ştie ce delincvență. Ceea ce vreau să spun este că sunt mulți designeri care combină elemente de design deja existente pentru a crea ceva original. Sunt destui de puțini cei care într-adevăr inventează ceva cu adevărat nou. De aceea veți fi puşi de multe ori în fața unei situații în care vedeți ceva interesant dar nu ştiți cum se face.

Exact acest mod de a aborda lucrurile îl voi folosi acum. Voi vorbi ca şi cum am văzut site-ul prima oară în viața mea şi vreau să creez ceva cât mai asemănător. Fac acest lucru pentru că situația aceasta nu apare numai când vedeți ceva nou la altcineva dar şi atunci când aveți un model în minte şi trebuie să-l transpuneți în HTML.

Pentru a realiza acest mic proiect am folosit şi anumite elemente de CSS. Despre CSS am mai vorbit, însă prea puțin. Voi încerca să explic ceea ce fac cât mai bine dar explicațiile vor fi concise şi la obiect pentru că CSS este un alt subiect care nu face obiectul acestui articol.

În primul rând începeți prin a crea un director separat care va conține toate fişierele site-ului. Apoi în acel director creați un director care va conține imaginile din site. Un nume bun pentru acesta ar putea fi "Imagini". Creați apoi un fişier "index.html" şi un fişier "default.css".

```
<html>
<head>
<title>Pagina linux360 pentru
articolul de HTML</title>
<link      href="default.css"
type="text/css"
rel="stylesheet">
</head>
<body>
...
</body>
</html>
```

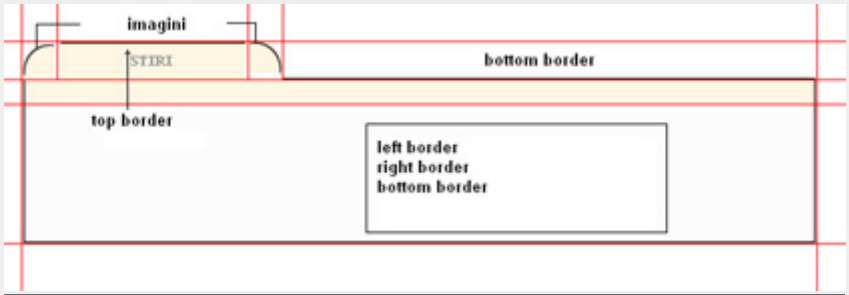
Acum să analizăm site-ul din imagine. Se observă trei zone mari: zona de header, zona meniului şi zona de conținut. Trebuie să facem în aşa fel încât să avem şi noi 3 astfel de zone independente una de cealaltă. Şi acest lucru îl putem realiza folosind un tabel.

Tabelul va avea două rânduri: primul va conține o celulă, iar al doilea rând va conține trei, celula din mijloc fiind o celulă cu rol de spațiere. Pentru ca prima celulă să ocupe acelaşi spațiu cu celulele din cel de-al doilea rând vom folosi atributul *colspan*.

```
<table width="800px" border=0
cellspacing=0 cellpadding=0>
<tr><td colspan=3>
<tr><td> 
<td                width="5px"
bgcolor="white"> 
<td> 
</table>
```

Am reuşit să împărțim spațiul în cele trei zone pe care le doream. Acum mai trebuie să punem şi header-ul. Problema header-ului ar putea părea simplă: punem o imagine în cadrul celulei şi gata. Nu e chiar aşa, deşi şi aceasta este o soluție. Se poate observa că partea din dreapta a header-ului este goală. Acea parte a fost lasată intenționat aşa. Acolo pot fi plasate butoane care să iasă în evidență: de exemplu un buton "About" care să facă trimitere la o pagină cu explicațiile necesare noului venit pentru a se acomoda mai uşor cu site-ul şi pentru a înțelege rolul acestuia.

Un template linux360 - acesta este rezultatul pe care trebuie să-l obținem



Datele necesare pentru a construi ultima parte din pagină

Dacă am putea să punem imaginea ca background, atunci toate problemele ar fi rezolvate, pentru că spațiul din interiorul celulei rămâne gol pentru a pune altceva în el. Mai mult, toate elementele din celulă ar fi automat plasate deasupra background-ului. Acest lucru este posibil folosind un style inline sau definind o clasă în fişierul *default.css*, pe care o atribuim ulterior celulei special creată pentru header.

```
<td style="background-image:
url(imagini/banner.jpg)"
height="129px" colspan="3">
```

Soluția anterioară este bine aleasă mai ales că nu o să mai avem nevoie de acest stil pentru a-l defini într-o clasă, dar dacă totuşi doriți să definiți stilul în fişierul *default.css,* am să dau în continuare şi această variantă.

```
.headercell {
background-image:
url(imagini/header.jpg); }
```

Scriind codul anterior în fişierul pentru stiluri *default.css* definim clasa *headercell* care se aplică oricărui element dar elementele definite în interiorul clasei nu sunt neapărat valide pentru orice element. Pentru a atribui celulei clasa *headercell* folosim parametrul *class*.

```
<td class="headercell"
height="129px" colspan="3">
```

Acum să ne ocupam de meniu. Aşa cum se poate observa, meniul este despărțit de linii punctate. Deasemeni, zona meniului şi zona din dreapta lui sunt despărțite tot de o linie punctată. Ce sunt şi cum se fac aceste linii punctate. Liniile punctate reprezintă de fapt o proprietate de border a unei celule, a unui tabel.

Pentru a atribui unei celule proprietatea de bordură într-o anumită extremitate a ei, trebuie să facem un stil css care să definească acest lucru.

```
.menucell { background-image:
url(imagini/menu.jpg);
border-right-width: thin;
border-right-style: dashed;
border-right-color: #CCCCCC;
background-repeat: no-repeat;
background-position: center
top; }
```

Liniile îngroşate se referă la proprietățile pentru linie întreruptă, restul adăugând o imagine de fundal. Această clasă a fost atribuită celulei inițiale de meniu. Ulterior, în această celulă a fost făcut un alt tabel al cărui celule aveau bordura de jos setată să fie o linie întreruptă.

```
<table border=0 cellspacing=0
cellpadding=0>
<tr><td width="150px">
<tr><td
class="bottomdashedborder">
 
<tr><td
class="bottomdashedborder">
<a href="stiri.html"
class="alink">    
  Stiri</a>
</table>
```

Clasa *bottomdashedborder* se face cam aşa.

```
.bottomdashedborder {
border-bottom-width: thin;
border-bottom-style: dashed;
border-bottom-color: #CCCCCC;
}
```

După cum se poate observa mai sus fiecare link are asociată o clasă care-i modifică aspectul, dar, şi mai important funcționalitatea (când deplasezi cursorul deasupra link-ului culoarea acestuia se schimbă). Pentru a implenta anumite proprietăți în funcție de alți parametri, trebuie să folosim o caracteristică a limbajului CSS, şi anume *selectorii. Selectorii* implementează potrivirea de şabloane (pattern matching). Vom folosi selectorii :link, :visited, :active, :hover:

```
.alink:link, .alink:visited
{
color: #888888;
text-decoration: none;
cursor: hand;
font-family: Arial,
Helvetica, sans-serif;
font-size: 16px; }
.alink:active, .alink:hover
{
color: #fb9312;
text-decoration: none;
cursor: hand;
font-family: Arial,
Helvetica, sans-serif;
font-size: 16px; }
```

În felul acesta puteți specifica o altă culoare a unui element html atunci când utilizatorul execută o anumită acțiune specificată de selector.

A rămas ultima parte din document, şi anume, zona de conținut. Aceasta nu are nimic nou față de ce am discutat până acum şi de aceea nu voi insista. Am oferit o imagine care să vă servească drept schemă, pe baza acesteia putând construi singuri ultima parte a paginii.

Sursele paginii cu toate resursele aferente pot fi descărcate de pe Internet. Găsiți adresa în căsuța cu resurse.

Resurse:

- www.linux360.ro/~cbidea/ht6.zip

Autor:

cristian.bidea@linux360.ro

Arta documentării software-ului

Silviu Foca

Acest articol naşte din intenția de a arăta câteva linii de ghidare simple pentru a reuşi documentarea software-ului pe care îl produceți, într-un mod cât mai simplu şi complet.

Documentația şi ciclul de viață al software-ului

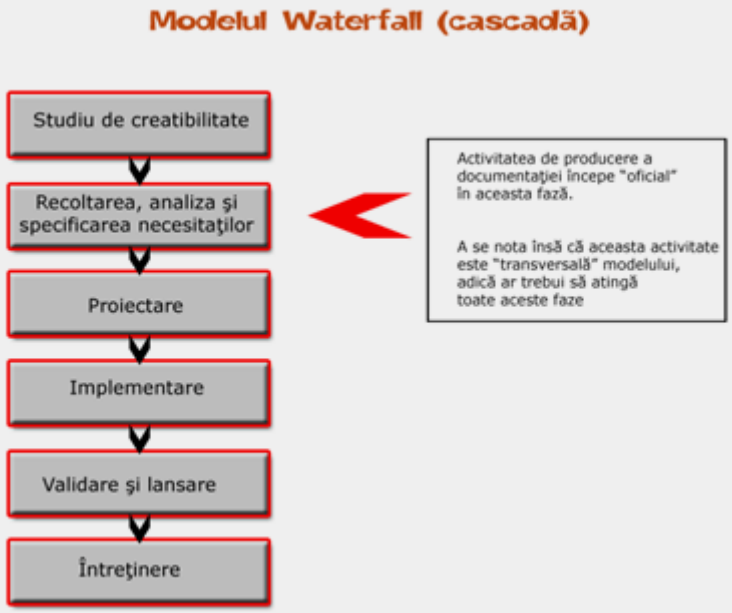
Faza de producție a documentației reprezintă una dintre cele mai importante faze în dezvoltarea unui software şi paradoxal este foarte des cea mai neglijată.

De câte ori nu am avut de a face cu un software pentru care nu există o documentație clară, completă şi la zi? Ingineria software-ului, sau mai bine zis branşa ingineriei definită de către IEEE ca "*disciplina tehnologică şi managerială ce priveşte producția sistematică şi întreținerea produselor software în timpii şi costurile prevăzute*", rezervă un loc special documentației. Nu la întâmplare, în istoricul model Waterfall ("*în cascadă*"), o schemă ce descrie ciclul de viață al unui software, faza de documentație începe "oficial" chiar din etapa de *Recoltare şi Analiză a Necesităților*, adică cu mult înainte de *Proiectarea şi Implementarea Proiectului*!

Documentația şi tipurile de folosință

Adeseori suntem predispuşi să ne gândim că documentația unui proiect coincide pur şi simplu cu manualul utilizatorului şi cu câteva note pe marginea subiectului. În realitate "documentație" este un termen ce include o serie întreagă de concepte şi nu se limitează doar la "folosirea din partea utilizatorului".

Fiecare proiect, de fapt, are cel puțin două tipuri de persoane interesate: cine îl produce şi cine îl fructifică. În prima



Modelul Waterfall

categorie intră proiectanții, analiştii, programatorii, betatester-ii etc. -- însa şi cine conduce dezvoltarea proiectului şi anume, managerii etc. În schimb în cea de-a doua clasă regăsim pe cei care fructifică proiectul, adică utilizatorul final. Din aceste considerente documentația trebuie să reflecte aceste niveluri diferite de cunostință şi "intimitate" cu software-ul. Utilizatorul foloseşte produsul însă nu poate sau nu este interesat de înțelegerea părților intime ale sistemului: lui îi ajunge sa cunoască cum sa interacționeze cu acesta. Din acest punct de vedere produsul este un "black box", adică o "cutie neagră" ce execută ceva, însă ale carei mecanisme de funcționare sunt necunoscute. Documentația în acest caz cuprinde manualul utilizatorului, unele note de utilizare, eventuale Frequently Asked Questions (FAQ - Întrebări Frecvente) ce abordează probleme uzuale etc. Tot ceea ce ajută utilizatorul să interacționeze corect cu produsul intră in acest tip de documentație.

În schimb o figură din prima categorie poate (şi trebuie) să cunoască produsul într-un grad mult mai mare decât simplul utilizator iar nivelul său de intimitate cu produsul poate să varieze în funcție de rolul său în dezvoltarea produsului. Asistăm deci la o trecere treptată de la viziunea de "black box" a utilizatorului către o viziune de "white box" a programatorilor, analiştilor etc. (adică o "cutie deschisă", cunoscută la nivel maxim).

Linii de ghidare în producerea documentației

Scrierea şi menținerea documentației nu este o problemă minoră: costă timp şi energie iar adesea rezultă plicticos (cel puțin în aparență). În mediul de lucru, în faza de implementare a proiectului, de obicei se pune accent pe "producerea a cât mai mult cod posibil" în loc de a se opri şi a documenta munca depusă până în acel moment. Această obişnuință este profund contraproductivă şi recomandat a fi evitată. În acest caz prima regulă poate fi definită astfel:

1. Scrieți documentația în manieră sistematică şi pas cu pas.

Avantajele sunt evidente:

- a scrie puțin din când în când e mult mai bine decât a scrie o singură dată mult (uneori chiar prea mult). Este absolut de evitat să aşteptați terminarea unui proiect pentru a scrie documentația deoarece va trebui sa faceți o muncă dublă: înainte de a scrie va fi necesar să vă forțați să vă amintiți ceea ce s-a făcut în trecut (tipic atunci când se uită comentariile în cod). Ajung doar câteva ore pentru a uita cel puțin o parte din ceea ce s-a produs până in acel moment!
- scrierea imediată a ceea ce se face ne poate permite individualizarea "din mers" a unor erori logice sau sintactice în ceea ce tocmai am produs. De exemplu, scriind un cod sursă, faptul de a trebui să încetinim ritmul pentru a revedea mental ceea ce am făcut şi să formulăm un comentariu, ne permite efectuarea unui prim control pentru a elimina eventualele erori (cel puțin cele mai evidente). În practică este ca şi cum am activa propriul nostru compilator mental (proces care in graba de a produce repede ceva, de obicei îl neglijăm).

Bineînțeles că sunt unele limite chiar şi în documentație. Regula de documentare pas cu pas a software-ului este eficientă atâta timp cât nu obstrucționează ceea ce trebuie să scriem! În linii maxime:

2. Este necesar să documentăm în manieră clară şi precisă ceea ce ar putea rezulta greu de înțeles pentru altcineva. Restul este opțional.

Da, exact, altcineva! Ideea este aceea de a considera fiecare proiect al nostru ca pe ceva ce va trebui citit de către altcineva (care probabil că nu dispune de toate calitățile, abilitățile şi cunoştințele nostre). Având permanent în minte acest mic truc va fi simplu de individualizat ceea ce este necesar a fi documentat şi ceea ce este facultativ, evitându-se astfel irosirea de timp şi energie. Printre altele în acest mod se evită riscul de supraproducție de documentație: prea puțină sau prea multă documentație apare negativ în ochii unei persoane ce are nevoie de o informație precisă. În realitate această tehnica este întotdeauna productivă (dacă aveți colegi ce trebuie să citească sau să modifice codul vostru vă vor fi recunoscători) şi foarte simplă de aplicat.

Un corolar destul de important la această regulă este acela de a:

- *documenta tot ceea ce se reține necesar, incluzând şi ceea ce pare banal sau superfluu pentru ochii noştri (însa nu ar putea fi aşa în ochii acelui faimos **altcineva**)*

Ceea ce nouă ne apare simplu sau chiar inutil poate reprezenta o problemă pentru alte persoane. Trebuie să evaluam atent cui i-ar putea ajunge în mâini documentația noastră şi să acționăm în consecință. De exemplu, dacă producem o documentație ce va fi folosită doar de programatori putem să folosim în linişte termeni tehnici şi abrevieri. Acelaşi lucru nu poate (şi nu trebuie) să se întâmple dacă respectiva documentație va fi folosită de către un non-programator, ca de exemplu un client total ignorant in domeniul tehnologiei şi al informaticii.

Un alt corolar util însă opțional este acela ce priveşte utilizarea surselor:

- *când este disponibilă documentație in plus, se va include şi aceasta. Dacă nu este posibilă includerea ei, se va cita sau se va face referire la sursă*

Este inutil să scriem documentație atunci când exista deja una, mai ales dacă este liber utilizabilă. Dacă nu este posibilă includerea acesteia direct (pentru că este pe un anumit site sau este protejată de o anumită licență) este oricum posibilă citarea sa şi eventual indicarea unei referințe.

Un exemplu concret: comentați un cod ce face referire la o anumită tehnică pentru care există documentație pe un anumit site web. Presupunem că această documentație poate fi vizionată însă nu inclusă direct în proiectul la care lucrăm (adică nu puteți face "cut`n paste" nici măcar citând autorul). În acest caz, văzând că nu avem alternativă, este oportună cel puțin indicarea URL-ului (puteți să semnați adresa chiar în fişierul sursă dacă v-ar putea fi de folos). Aceasta ar putea folosi atât acelui faimos "altcineva" care ia în mână software-ul vostru cât şi vouă înşivă: dacă aveți probleme şi doriți să o vizionați din nou veți avea link-ul la îndemână.

Acum să ne oprim asupra redării documentației, adică tehnicile ce facilitează înțelegerea ei. O primă regulă în acest context este sintetizabilă în:

3. Dacă este posibil, este de preferat reprezentarea conceptelor cu scheme şi diagrame decât descrierea lor prin cuvinte.

În afara de a ne ajuta pe noi înşine să înțelegem cum sa procedăm în faza de dezvoltare (evidențiind erori şi anomalii de orice tip şi formă) dacă sunt scrise adecvat, permit o înțelegere mai uşoară chiar şi din partea celor ce nu dispun de un grad de cunoştință similar cu al nostru. În afara de aceasta, puterea expresivă a unei diagrame este net superioară de cea a cuvintelor: chiar şi o simplă mâzgâlitura este în masură de a defini şi descrie "vizual" concepte ce ar necesita zeci şi zeci de rânduri de text!

O diagramă/schemă pentru a fi inteligibilă trebuie să fie:

- *corectă*: trebuie să oglindească exact ceea pentru ce a fost desenată
- *completă*: tot ceea ce este necesar trebuie să fie redat. O parte lipsă echivalează cu un "gol de memorie" în proiect cu toate consecințele sale
- *inteligibilă*: o mâzgâlitură in unele cazuri este optimă, însă in altele nu. Daca se descrie un concept complex, este nevoie de o diagramă ce nu va constrânge utilizatorul să înnebunească între săgeți şi blocuri explicative. De asemenea şi dimensiunile sunt fundamentale: concepte complexe pot necesita diagrame destul de voluminoase pentru a putea fi citite. Evaluați dacă este cazul de a diviza conceptul reprezentat în mai multe sub-diagrame.
- *minim*: este necesar să evitați duplicate, redundanțe etc. ce pot duce

```
[ ... ]

//******** INCEPUTUL CODULUI ******************

/************
* HEADERS *
************/
include <stdio.h>

/******************
* PROTOTIPURI *
******************/
int returneazaStatus (void);

/********************
* IMPLEMENTARE *
********************/

/**
* returneaza statusul curent al sistemului (sub forma de cod)
*/
int returneazaStatus (void)
{
      [ ... ]
}
[ ... ]

//*********SFARSITUL CODULUI******************
```

la neînțelegeri sau înțelegeri defectuoase a conceptului.

Există diferite tipuri de diagrame (ex. de flux, în blocuri, de secvență etc.): fiecare dintre acestea este potrivită unor contexte diferite, evidențiind diferite puncte de vedere şi posedând o anumită putere explicativă.

Dacă doriți producerea de documentație de calitate vă recomand sa dați o privire la diagramele limbajului de modelare UML, de acum fiind un standard internațional pentru producerea de diagrame. Din păcate UML e un limbaj destul de complex şi ar ocupa singur diverse articole (ca să nu spun carți): totuşi vă îndemn către numeroasele ghiduri şi tutoriale disponibile gratuit pe Internet.

O a doua tehnică, aşa-zis vizuală, este complementară cu precedenta:

3. Recurgeți la o formatare clară şi accesibilă tuturor.

Puteți scrie cea mai bună documentație din lume însă dacă utilizați un stil "plat", nu veți facilita înțelegerea nici a utilizatorului şi nici a voastra. La urma urmei şi ochiul vrea partea sa, iar programatorii/analiştii sunt dotați şi ei cu aşa ceva.

Această regulă aparent banală, implică adoptarea unei metodologii bine definite: când se alege o anumită formatare (culori specifice, fonturi cu proprietăți particulare etc.) trebuie menținută omogenă în toată documentația. Nimic nu este mai iritant decât consultarea unui manual în care fiecare capitol conține diferite formatări: de obicei creierul nostru interpretează schimbul de formatare/stil/temă ca o schimbare logică, adică un nou "bloc conceptual dotat de o semnificație proprie", diferită de precedentele. Să fim înțeleşi: tot ceea ce am spus nu implică obligativitatea indicarii explicite a semnificației fiecărei formatări, metoda tipică a cărților (clasica asociere a stilului cursiv cu citatele, a stilului monospace cu părțile de cod etc.). Ajunge să decidem un stil bine definit şi să-l aplicăm, dacă este posibil, la toată documentația. Puteți sa adoptați un mecanism (derivat de la web) ca de exemplu o foaie de stil, un fişier `.css` (sau `.xsl`dacă preferați XML), ce poate fi aplicat tuturor paginilor HTML dintr-un sit: în practică este destul să scrieți oportun textul apoi să aplicați foaia de stil pentru a obține o documentație omogenă. În acest caz puteți face experimente grafice fără a trebui sa înnoiți pagină după pagina toată documentația. Ultima metodă, foarte placută, însă care cere o anumită grijă, este folosirea limbajelor/formatelor de paginare avansată precum `TeX`/`LateX`.

Acum să incercăm aplicarea acestor linii de ghidare în timpul fazei de implementare a software-ului, adică în timpul scrierii codului sursă.

Un caz particular: Documentația codului sursă

Dezvoltarea şi menținerea documentației codului sursă a unui proiect este un punct fundamental şi contribuie enorm la calitatea proiectului. Este important de evidențiat că documentația unui cod sursă include atât comentariile şi notele din fişierul sursă cât şi diagramele, schemele, notele etc. ce "însoțesc" fişierele sursă.

În acest context, din motive de simplificare, considerăm că documentația codului se rezumă la comentariile (şi notele) din fişierul sursă.

Regula fundamentală este:

File Edit Search View Tools Options Language Help

```
/*****************************************************************************
*PROIECT:        testCompilator                                              *
*FISIER:         main.c                                                      *
*DESCRIERE:      fisierul principal al proiectului, contine functia main()   *
*REFERINTE:      cod de exemplu inspirat de codul de la                      *
*               http://www.coduri-libere.org/articol/exemplu.html           *
*               (liber utilizabil, licenta GPL)                             *
*ULT. MOD.:      14.09.2004 08.00 (autor: fdisk)                             *
*AUTOR:          Silviu Foca ( silviu.foca@linux360.ro)                      *
*COPYRIGHT:      (c) 2004 by Silviu Foca                                     *
*LICENTA:        lansat in termenii licentei GNU, GPL v2.0                   *
*****************************************************************************/
```

- *scrieți orice lucru în maniera cea mai clară şi mai simplă cu putință.*

Evitați fraze lungi pentru exprimarea unor concepte: cateva cuvinte pot fi îndeajuns pentru definirea unor concepte simple. Pentru concepte complexe puteți scrie note referitoare sau puteți introduce referințe la fişiere şi resurse externe fişierului sursă.

Si următoarele reguli de bază sunt la fel de simple:

- *fiecare fişier trebuie să conțină în antet cel puțin: unele informații de bază despre autor (nume, email etc.) şi despre proiectul în care este inclus fişierul de față precum şi o descriere minimă a scopului acestuia din urmă.*

Este de dorit să indicați şi licența al cărei subiect este fişierul în cauză (e.g. GPL) şi/sau o indicație despre cine deține drepturile asupra acelui fişier (e.g. firma pentru care lucrați).

Dacă la proiect participă mai multe persoane şi gestionarea fişierelor sursă nu se face cu ajutorul unui sistem de control al reviziilor precum CVS (adică un program pentru gestionarea eficientă a unui proiect, ce permite mai multor persoane să lucreze într-un mod sigur chiar şi asupra aceloraşi fişiere, simultan), poate fi utilă şi indicarea persoanei ce a efectuat ultima modificare şi când a fost efectuată aceasta.

Multe instrumente de dezvoltare (e.g. KDevelop, Sun+ONE etc.,...) furnizează deja sabloane (adică fişiere de model) gata pentru a fi folosite, conținând scheletul de completat cu informațiile mai sus citate. Dacă folosiți un program ce nu oferă această funcționalitate va puteți crea automat un "model". Trebuie doar să creați un fişier cu scheletul şi apoi să setați atributul read-only (doar citire) pentru acest fişier: în acest mod puteți să-l utilizați ca bază pentru a scrie codul vostru însă va fi necesar să-l salvați cu alt nume astfel conservându-se modelul. Această fază poate fi uşurată prin intermediul unui program denumit Doxygen.

```
[ ... ]
if (...)
      persoana.selectioneaza(obiect);     //persoana _trebuie_ sa selectioneze un obiect
      persoana.cumpara(casa);             //persoana _trebuie_ sa cumpere o casa
[ ... ]

[ ... ]
if ( ... )
      //persoana _trebuie_ sa selectioneze un obiect
      persoana.selectioneaza(obiect);

      //persoana _trebuie_ sa cumpere o casa
      persoana.cumpara(casa);
[ ... ]
```

Exemplu de cod comentat

În exemplul din figură informațiile de catalogare au fost puse într-o casetă, un chenar format din asteriscuri, ce permite individualizare şi distinge vizibil această parte de codul real.

Regula poate fi exprimată prin:

- *folosiți casete, linii de separare (formate şi acestea din simboluri), şi orice altă metodă de subdividere "vizuală" când doriți să distingeți imediat porțiuni de cod logic separate*
- *După cum am zis şi mai sus, este important să comentați codul mai ales în locurile unde, după parerea voastră, ar crea dificultați pentru alte persoane ("faimosul" altcineva).*

Folosiți-vă de comentarii şi pentru a indica situații particulare sau pentru a evidenția comportamente "non-standard". Fiecare limbaj oferă simboluri particulare pentru indicarea zonelor de comentariu. Adoptați un stil precis şi mențineți-l de-a lungul întregului proiect.

Dacă scrieți cod C++ de exemplu, puteți decide intre a pune comentariile pe acelaşi rand cu codul sau pe randul precedent. După cum se observă şi din figura 4, comentariile în primul caz apar in partea dreaptă şi sunt încolonate între ele. În schimb în cel de-al doilea exemplu, comentariile preced liniile de cod; pentru marirea lizibilitații a fost introdusă o linii goală ce duce la apariția vizibilă a două micro-blocuri de cod. Acestea sunt alte două ingrediente ce măresc (şi nu puțin) lizibilitatea codului pe care îl creați.

Concluzii

Sper că acest articol nu vi s-a părut plicticos şi, mai ales, sper că mi-am atins scopul pe care mi l-am propus: acela de a vă atrage atenția, stimulându-vă să aprofundați această temă care este foarte des uitată (nu prea am găsit pe Internet sau prin cărțile de specialitate abordări serioase asupra acestei teme).

Resurse:

- http://www.omg.org/uml/

Autor:

silviu.foca@linux360.ro

Programarea şi uneltele ei

Costin Gamenț

Ce înseamnă programare? Dacă citim în *Dicționarul Enciclopedic*, aflăm ca **program (< fr. germ., it.; {s} pro- + gr. gramma ¨scriere¨) = Ansamblu complet de instrucțiuni ce pot fi executate de un calculator pentru rezolvarea unei probleme date**, adică exact ceea ce a facut maşinile noastre atât de populare -- nu trebuie să schimbi circuitele ca să rezolvi alt fel de problemă, trebuie doar să ştii cum să comunici microprocesorului noul set de date şi felul în care acestea vor fi modificate.

Ca în orice fel de dialog, şi în "discuția" cu procesorul este imperios necesar să se vorbească aceeaşi limbă. Calculatorul înțelege un set bine definit de comenzi, denumit *opcodes* (sau *coduri de instrucțiune*, într-o traducere inexactă) şi care, pentru arhitecturile x86 (compatibile Intel) sunt în binar. O arhitectură este formată din procesoare care, chiar dacă sunt produse în alți ani, de alte firme sau au anumite opcode-uri noi, au un set de bază de comenzi identice. Un exemplu îl reprezintă arhitectura x86, care are la bază opcode-urile lui i8086, în implementările sale de la Intel, AMD, Texas Instruments sau Cyrix. Altă arhitectură este cea promovată de Apple, diferită de x86 — de aceea nu se pot rula programe scrise pentru Mac pe PC-uri (şi viceversa). Din păcate, oamenii nu gândesc în binar şi este destul de greu să scrii cod direct folosibil de procesor. Primul pas a fost inventarea limbajului de asamblare.

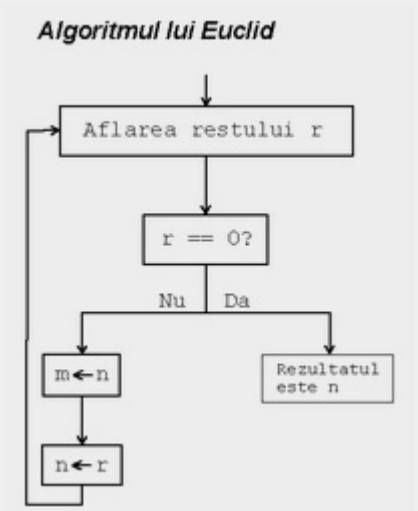
Limbajul de asamblare, deşi este suficient de greoi, este cât de cât inteligibil deoarece foloseşte abrevieri ale unor cuvinte din limba engleză (numite *mnemonici*) pentru a denumi opcode-urile respective. Normal că a fost necesară scrierea unui "traducător" de limbaj de asamblare în binar denumit, din motive evidente, asamblor. În realitate, asamblorul are ca rezultat un fişier binar într-un format cunoscut de sistemul de operare, sistem de operare care îl transformă în continuare în format cunoscut de procesor.

Odată cu dezvoltarea sistemelor diferite de operare, s-a constatat că un program scris în limbaj de asamblare era specific unei anumite arhitecturi şi unui anumit sistem de operare. În căutarea portabilității s-au inventat alte limbaje de programare, denumite de *nivel înalt*, care au drept scop tocmai această portabilitate. În teorie, dacă există un compilator (adică un program ce ştie să transforme din limbaj de nivel înalt în limbaj de asamblare) pentru o anumită platformă (prin platformă înțeleg o combinație de arhitectură şi sistem de operare) se poate folosi un program scris într-un limbaj de nivel înalt. Bineînțeles că în practică lucrurile nu stau tocmai aşa, din cauza complexității uimitoare la care au ajuns atât sistemele de operare, cât şi hardware-ul pe care acestea rulează. Exemple de astfel de limbaje de nivel înalt sunt cele de care unii dintre noi ne-am lovit, sau ne vom lovi în liceu — *ANSI C* şi *Borland Pascal*.

Voi prezenta în continuare o listă ceva mai completă cu paşii urmați de codul nostru sursă până la execuție (vezi şi imaginea alăturată):

a. **preprocesare** — se verifică sintaxa şi se execută directivele de preprocesor (în cazul C, aici se includ fişierele antet `.h`). Dacă totul decurge corect, rezultatul este tot un fişier sursă, dar primenit şi gata de citire pentru compilator
b. **compilare** — fişierul rezultat din preprocesare este transformat în limbaj de asamblare
c. **asamblare** — rezultatul compilatorului (codul de asamblare) este transformat într-unul din formatele cunoscute de asamblor, oricare din acestea fiind binar
d. **editare de legături** — dacă în programul nostru am folosit rutine dintr-una sau mai multe biblioteci de programare, aici se vor insera fie codul (dacă biblioteca este legată static), fie referința rutinei respective (pentru legarea dinamică)
e. la **execuție**, sistemul de operare interpretează fişierul binar rezultat în urma legării, transformându-l in opcode-uri pe care le inserează direct în procesor.

Până acum a fost o prezentare tehnică, dar este cunoscut faptul că o persoana care ştie C nu este neapărat un programator iscusit. Putem să ne gândim la un calculator ca la un abac deosebit de rapid. Comparația este forțată bineînțeles (nu am auzit de abac cu magistrală de date), dar poate constitui o analogie utilă în procesul de învățare. În fond, orice calculator de astăzi "ştie" să execute operații aritmetice de bază la o viteză uluitoare, dar felul în care aceste operații sunt combinate pentru a produce un rezultat util în rezolvarea unei probleme date depinde numai şi numai de programator şi de felul în care este realizat programul. Iar acest "felul în care este realizat programul" se traduce prin *algoritm*.



Un algoritm ilustrat

Printr-un algoritm se înțelege un set finit de reguli care indică o secvență de operații ce au ca rezultat soluția într-un anumit tip de problemă, cu alte cuvinte, o metodă care să funcționeze de fiecare dată şi care să furnizeze rezultatul corect de fiecare dată.

Se pot sesiza imediat cinci caracteristici foarte importante ale oricărui algoritm:

1. *caracterul finit* – indiferent de natura problemei, nu este acceptabil un algoritm cu un număr de paşi infinit. Deosebit de mare – da (a se vedea si pasul *5*.), infinit – niciodată
2. *definit* – fiecare pas al unui algoritm trebuie bine documentat şi bine stabilit, urmând a fi exprimat într-un limbaj deosebit de strict (limbajul de programare), ce nu admite jumătăți de măsură
4. *intrările* – orice algoritm (sau program de calculator) are un set de date de intrare variabile ce sunt furnizate prin metode specifice (citire dintr-un fişier, introducere manuală etc.). Aceste date de intrare (inputs în limba engleză) pot fi furnizate fie la începutul rulării, fie dinamic, pe parcursul execuției
5. *ieşirile* – bineînțeles reprezintă ceea ce s-a dorit de la început – soluția (sau soluțiile) problemei
6. *eficiență* – spuneam la pasul *1*. că un algoritm trebuie să aibă caracter finit. În practică, este de aşteptat ca un algoritm să se execute nu numai într-un număr finit de paşi, ci chiar în numărul minim de paşi. De asemenea, ideal ar fi ca orice algoritm să poată fi testat cu uşurință pe hârtie, astfel asigurându-se folosirea unor metode simple, deci rapide (a se citi operații aritmetice de bază).

După atâta teorie, n-ar strica şi un exemplu. Unul dintre "clasici" este *Algoritmul lui Euclid*, învățat în clasele de gimnaziu cu respect la aflarea celui mai mare divizor comun a două numere întregi (pozitive) `m` şi `n` (vezi şi schema):

1. `r` = restul împărțirii lui `m` la `n`
2. dacă `r = 0`, algoritmul se încheie cu rezultatul `n`
3. se atribuie valoarea `n` lui `m`
4. se atribuie valoarea `r` lui `n` şi se revine la pasul *1*.

Am descris aspectul tehnic, am descris aspectul teoretic, dar am omis poate cel mai important – practica. În realitate, tot procedeul complicat de compilare (până la legare) este executat de un singur program (un exemplu elocvent este `gcc`-ul cel de toate zilele), iar legarea este făcută tot de compilator, prin apelarea altui program (în lumea Linux se foloseşte preponderent `ld`). Spuneam puțin mai sus că o persoană care ştie un limbaj de programare nu este neapărat un programator adevărat, iar acum este momentul să argumentez: dacă se poate ca respectivul programator să nu ştie cum se realizează procesul de compilare, nu este admis să nu ştie cum se produce un algoritm eficient pentru că, în fond, limbajul de programare nu este decât modalitatea prin care se descrie algoritmul. Am văzut foarte mulți oameni care încep rezolvarea unei probleme gândind în termeni de `ANSI C`, spre exemplu. E ca şi cum te-ai gândi la cum să faci ceva, dar fără să ştii ce vrei să faci.

Fără a emite pretenția că documentul de față tratează exhaustiv subiectul propus şi fără a fi încercat o abordare extraordinar de savantă, sper doar că am reuşit să fac puțină lumină în ce inseamnă conceptul de programare şi care sunt direcțiile de urmat de un începător.

Resurse:

- *Arta programării calculatoarelor*, Donald E. Knuth

Autor:

costin.gament@linux360.ro

rsync - soluție de backup

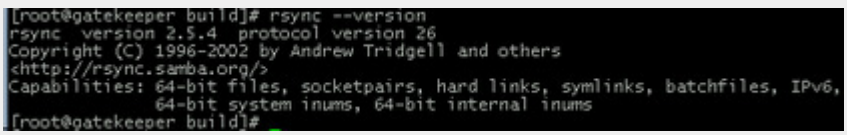
Daniel Secăreanu

`rsync` este un utilitar de linie de comandă folosit pentru a sincroniza fişiere între două calculatoare, fie ele servere de ftp, web sau de alt tip. `rsync` însă, poate fi folosit şi ca unealtă de backup pentru servere de fişiere sau chiar şi pentru calculatorul de acasă. `rsync` este o unealtă facilă ce poate fi folosită pentru backup manual sau chiar şi automat prin intermediul `cron` de orice utilizator. În rândurile următoare vom vedea cum se poate face backup pe o unitate de disc internă sau externă şi chiar pe un alt calculator aflat la distanță.

În primul rând trebuie verificat dacă avem pachetul `rsync` instalat pe calculator. Pentru utilizatorii sistemelor bazate pe managerul de pachete `rpm`, acest lucru se face tastând comanda

```
$ rpm -qa | grep rsync
rsync-2.6.2-8.9;
```

sau pur şi simplu, pentru utilizatorii sistemelor bazate pe surse, se scrie în consolă `rsync --version` comanda generând un output similar cu cel din figură.



Dacă această comandă returnează eroarea:

```
$ rsync --version
-bash: rsync: command not found
```

atunci înseamnă că trebuie instalat pachetul `rsync`. Acesta poate fi găsit ca pachet inclus în majoritatea distribuțiilor sau poate fi descărcat de pe Internet de la adresa de la resurse. Din motive de securitate, este recomandat să vă asigurați de faptul că versiunea folosită este mai mare decât 2.6.0.

Pasul următor este să stabilim unde vom face backup şi ce fel de backup dorim să facem. Un backup complet crează o copie identică a hard disk-ului, oferind o metodă rapidă şi sigură de restaurare a sistemului şi a datelor în cazul unui dezastru. Cel mai adesea, aceste tipuri de backup durează foarte mult, consumă extrem de mult spațiu şi nu sunt neapărat necesare.

Atunci când faceți totuşi aceste backup-uri, asigurați-vă că folosiți opțiunea `--exclude`, deoarece există anumite directoare ce nu trebuie salvate, cum ar fi `/proc`, de exemplu.

Backup-urile parțiale de sistem sunt mult mai rapide şi mult mai eficiente, deoarece sunt salvate doar datele cele mai importante sau părțile de sistem cele mai importante, cum ar fi `/etc` sau `/home`, de exemplu. Directoarele de sistem, `/usr` sau `/var`, pot fi reinstalate cu uşurință, şi adesea nu trebuie salvate. În cazul în care doriți să arhivați şi mesajele de sistem sau stocați e-mail-ul în `/var/spool/mail`, atunci este recomandat să salvați şi directorul sau partiția `/var`.

Acum că am stabilit cam ce ar trebui salvat, haideți să vedem unde salvăm toate aceste informații. Răspunsul la această întrebare e limitat doar de imaginația şi resursele voastre. Datele pot fi salvate pe o partiție separată, pe o unitate de bandă, pe un cd sau chiar prin rețea pe un alt calculator. Când ar trebui făcute aceste backup-uri? În funcție de importanța datelor, salvarea lor ar trebui făcută periodic şi cât mai frecvent, folosind scripturi de shell, `cron` sau alte unelte disponibile. În articolul de față vă vom prezenta doar salvarea datelor pe unități de disc locale.

Cum folosim `rsync`? Este destul de simplu. Conceptul de bază este următorul:

`rsync -a /sursă /destinație.`

Această comandă copiază directorul sursă în directorul destinație ca şi cum am fi dat comanda `cp /sursă /destinație`. Spre deosebire de `cp`, `rsync` se foloseşte de propriile algoritme pentru a verifica dacă există diferențe între cele două directoare. Prin acest algoritm, `rsync` poate copia doar fişierele ce s-au modificat de la ultima operație, facând astfel posibilă operația de backup incremental, o metodă extrem de rapidă de a aduce la zi un director, fără a fi nevoie să copiem întreg conținutul acestuia, ci doar fişierele ce s-au modificat între timp.

Dacă totuşi se doreşte copierea completă a directorului, atunci poate fi folosită opțiunea `-delete`, aceasta cauzând `rsync` să şteargă conținutul directorului `/destinație` înaintea copierii conținutului directorului `/sursă`. De fapt, acest flag şterge fişierele din `/destinație` ce nu există şi în directorul `/sursă`. Acest lucru ne asigură că backup-ul este o copie fidelă a datelor noastre, în eventualitatea că am şters un fişier inutil şi am vrea ca acesta să fie şters şi din backup.

În general este o idee bună să avem mai multe copii ale datelor salvate în diferite zile, pentru a ne asigura că nimic nu se poate întâmpla ca să rămânem fără informațiile de valoare. Acest lucru se poate face extrem de uşor prin intermediul unui mic script de shell.

Atunci când ne pregătim să facem operațiunea de backup trebuie să ne asigurăm că există destul spațiu pe discul destinație, fie el un disc secundar din calculator, o unitate USB externă, o unitate i.Link sau un alt calculator de pe rețea.

Pentru a verifica spațiul ocupat de directorul în care ne aflăm, putem folosi comanda `du -sh ./`. Pentru a afla cât spațiu avem disponibil pe discurile montate pe sistem, putem folosi comanda `df -h`.

Dar haideți să creem un script pentru salvarea datelor de pe o sursă pe o destinație. Folosind editorul preferat de text, copiați codul de mai jos într-un fişier pe care îl puteți denumi `backup.sh`. Hotărâți-vă ce fişiere şi directoare vreți să salvați şi modificați în script variabila `SOURCES`. Similar, decideți unde doriți să salvați datele respective şi modificați variabila `TARGET` pentru a reflecta acest lucru.

```
rsync-backup.sh      [----]  0 L:[  1+ 0   1/ 18] *(0    / 326
#!/bin/bash

# change to target directory holding backups
cd /target_directory

# move oldest backup to temporary name
mv day3 oldest

# increment backups, rotate oldest backup
# to today's
mv yesterday day3
mv today yesterday
mv oldest today

# update todays backup
cp -al yesterday/. today
rsync -a --delete source/ today/
```

Salvați scriptul şi setați permisiunile de execuție folosind comanda `chmod +x backup.sh`

În momentul de față sunteți gata pentru a salva primele date. Acest lucru se poate face extrem de uşor scriind comanda `./backup.sh`

Va dura ceva timp pentru ca procesul să se încheie, în funcție de cantitatea de date pe care doriți să o stocați. Dacă vreți ca acest proces să ruleze în background, adăugați `&` după comandă.

În acest fel puteți utiliza în continuare consola în care lucrați. Primul backup va fi cel mai lent, următoarele făcându-se mult mai rapid, deoarece doar fişierele modificate vor mai fi copiate (în cazul în care nu folosiți opțiunea `--delete`). În cazul în care ceva nu este în regulă, puteți oricând opri procesul de backup apăsând combinația de taste `CTRL+C`. În cazul unei copieri complete şi fără erori, veți avea o copie fidelă a datelor din `/sursă` în directorul `/destinație`.

Dacă procesul a rulat fără erori, acesta poate fi automatizat foarte uşor folosind serviciile puse la dispoziție de `cron`. Ne editâm propria listă de job-uri temporizate folosind comanda `crontab -e`. În fişierul care se deschide adăugăm linia de comandă ce va rula procesul de backup când avem nevoie, folosing sintaxa clasică `crontab` (vezi `man 5 crontab` pentru detalii). Astfel, adăugând linia:
`0 4 * * * /path/to/backup.sh`
vom executa un proces de backup în fiecare zi la ora 4 dimineața. Odată adăugată linia în fişier, acesa se salvează.

Cam asta a fost tot. După cum ați observat probabil, `rsync` este o unealtă extrem de facilă şi în acelaşi timp extrem de puternică pentru a face backup atât pe desktop cât şi pe server-e.

```
rsync-backup.sh      [----]  5 L:[  1+39  40/ 42] *(1064/109
#!/bin/sh
# backup.sh -- backup to a local drive using rsync

# Directories to backup. Separate with a space. Exclude
# trailing slash!
SOURCES="/home/wendy /home/daisy /var/mail"

# Directory to backup to. This is where your backup(s)
# will be stored.
TARGET="/mnt/usb-harddrive/backup/"

# Your EXCLUDE_FILE tells rsync what NOT to backup. Leave
# it unchanged if you want to backup all files in your
# SOURCES. If performing a FULL SYSTEM BACKUP, i.e. your
# SOURCES is set to "/", you will need to make use of
# EXCLUDE_FILE.
# The file should contain directories and filenames, one
# per line.
# An example of a EXCLUDE_FILE would be:
# /proc/
# /tmp/
# /mnt/
# *.SOME_KIND_OF_FILE

EXCLUDE_FILE="/path/to/your/exclude_file.txt"

# Comment out the following line to disable verbose output
VERBOSE="-v"
############################

if [ -f $EXCLUDE_FILE ]; then
    EXCLUDE="--exclude-from=$EXCLUDE_FILE"
fi

for source in $SOURCES; do
    if [ ! -d $TARGET/$source ]; then
        mkdir -p $TARGET$source
    fi
    rsync $VERBOSE $EXCLUDE -a --delete \
     $source/ $TARGET/$source/
done
```

Autor:

daniel.secareanu@linux360.ro

Tips & tricks

KDE Tips'n'Tricks

Dacă ați deschis vreodată o aplicație în KDE şi fereastra a fost prea mare pentru a ajunge la butonul de minimizare, puteați ține apăsată tasta ALT şi apoi, făcând click dreapta, să mişcați fereastra.

Au fost cazuri când după un upgrade la KDE 3.1, textele de sub iconițele de pe desktop au ajuns la o distanță maricică (e.g. 1 cm) de iconițe. Această problemă poate fi rezolvată făcând click dreapta pe desktop, apoi *Configure Desktop -> Behavior* iar în final deselectând căsuța de la *Images* din secțiunea *Show Previews For*.

Pentru a-l face pe Kmail să verifice mail-urile noi la startup, faceți click dreapta pe iconița Kmail-ului, alegeți *Preferences/Properties* iar la *execute/command* schimbați comanda `kmail --caption "%c" %i %m` în `kmail --check -caption "%c" %i %m`.

Pentru a alege care browser să se deschidă când faceți click pe un link, în KDE Control Panel la *File Browsing -> File Associations* alegeți *Text*, găsiți *HTML* şi faceți click pe *Add*. În continuare adăugați browser-ul care doriți mutându-l la capul listei pentru a-l pune ca şi browser implicit. Faceți click pe browser-ul pe care tocmai l-ați adăugat, apoi pe *Edit* şi în fereastra nou apărută, la *execute* adăugați %u la sfârşitul comenzii. De exemplu, pentru Mozilla ar fi: `/usr/bin/mozilla %u`.

Pentru a spune în care din cele n desktop-uri virtuale să se deschidă o aplicație pe care tocmai o porniți din modul text, puteți folosi kstart. Exemplu:
`kstart --desktop 3 konsole`
va porni o sesiune de konsole în desktop-ul 3.

Konqueror tips:

Konqueror are câteva web shortcuts care fac navigarea şi totodată lucrul la computer mai rapid şi uşor. Iată câteva din cele care vă vor fi mai folositoare. Nu uitați că le puteți modifica sau adăuga altele din *Settings -> Configure Konqueror -> Web shortcuts* (sau *Enhanced browsing* în versiunile mai vechi). Se introduc în bara de locație (*Location Bar*) din Konqueror.

`dict:moonstruck` vă va da definițiile în engleză pentru *moonstruck*

Glosar comenzi

ypdomainname

schimbă (setează) numele de domeniu NIS pentru maşina curentă

---

zcat

decomprimă intrarea standard (sau argv[1]) conform algoritmului gzip şi scrie rezultatul la ieşirea standard.

---

addpart <dispozitiv> <numar partitie> <inceput> <lungime>

adaugă în mod automat o partiție cu parametrii specificați pe dispozitivul de stocare dat ca argument.

---

arping [-I]

permite aflarea adresei fizice a unui nod IP.

---

blockdev <optiuni> [<dispozitiv>]

permite manipularea diverselor proprietăți comune tuturor dispozitivelor de tip bloc (de obicei discuri). De asemenea, poate fi folosit pentru a forța recitirea tabelei de partiții a unui astfel de dispozitiv.

---

cardctl <comanda>

foloseşte la controlul conectorilor PCMCIA şi a funcționării modulelor omonime.

---

consoletype

întoarce informații privind natura tipului particular de TTY de pe care a fost apelată comanda.

---

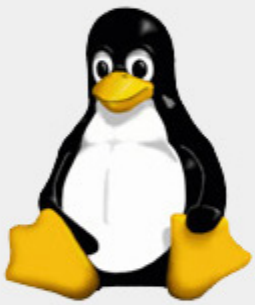
ctrlaltdel hard|soft

modifică comportamentul sistemului la apasarea celebrei combinații de taste.

---

delpart <dispozitiv> <numar partitie>

omologul lui addpart, şterge partiția dată de pe dispozitivul dat ca argument, în mod automat.



"Nu mă tem de calculatoare. Mă tem de lipsa lor."

Isaac Asimov

linux360 - numărul 09 - septembrie 2004