LABVirt – basic components

Asist. Codrin NISIOIU, asist. Cătălin SILVESTRU
Catedra de Informatică Economică, A.S.E Bucureşti

This article presents some aspects about a project integrated in a national research and development project "Information society" (INFOSOC) financed by Romanian Government. A iLab called LABVirt is available for use by Universities, college, high-school.. The goal of this paper is to present the main components of LABVirt, Romanian solution for remote lab access.

***Keywords**: iLab, remote lab access, client-server arhitecture, client applet*

Introducere

Sistemul educaţional prin Internet reprezintă un concept utilizat pe plan naţional în activitatea instituţiilor de învăţământ de toate gradele. Alinierea la ultimele tendinţe pe plan mondial în domeniul organizării şi funcţionării reţelelor de calculatoare după modelele de tip intranet/internet(adică revenirea de la modelul informatic în care datele şi prelucrările erau distribuite în mod “egal” în toată reţeaua, la arhitecturi cu servere puternice, deservite de soluţii software şi hardware specializate, care să ofere informaţii unor “clienţi” cât mai diverşi, fie aceştia calculatoare fixe sau mobile), creează premizele integrării cât mai apropiate în societatea informatică a viitorului [1] . Articolul va realiza descrierile arhitecturilor aplicaţiei client, respectiv server.

Descrierea aplicaţiei client

În figura 1 este prezentată comportarea dinamică a APPLET-ului client printr-o succesiune de stări ale acesteia. Fiecare stare corespunde unui nou comportament functional al aplicatiei.

Applet-ul iniţializează aplicaţia client (Starea 1) cu o prezentare generală a laboratorului virtual, prin care i se explică utilizatorului modul de folosire al aplicaţiei, modul în care se navighează în interiorul laboratorului virtual, ceea ce trebuie să facă pentru a duce la bun sfârşit lucrarea de laborator virtual si permite utilizatorului alegerea unei categorii de laboratoare din cele disponibile.

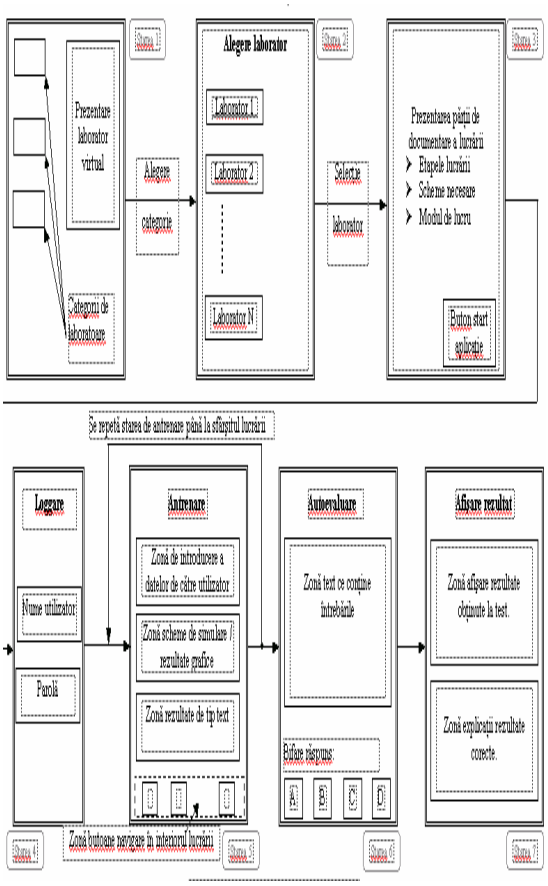

Fig.1. Descriere aplicaţie client

După ce clientul selectează o categorie, aplicaţia client trece în Starea 2 în care utilizatorului i se prezintă o listă de lucrări din care trebuie să aleagă laboratorul pe care trebuie să-l efectueze. Utilizatorul poate să aleagă orice lucrare doreşte, chiar dacă a mai parcurs-o într-o dată anterioară, sau începe o lucrare nouă. De asemenea, clientul poate efectua lucrările de laborator în ordinea dorită, nu există restricţii de acest fel.

În urma acestei selecţii, se trece în Starea 3, unde se presupune că utilizatorul este deja

familiarizat din Starea 1 cu modul de efectuare a unei lucrări de laborator virtuale. În acest moment, utilizatorului îi sunt prezentate numai informațiile legate de evoluția lucrării de laborator, cum ar fi etapele lucrării, schemele necesare şi modul de lucru. După ce utilizatorul a parcurs toate aceste informații, el poate să acceseze partea de simulare a lucrării, aplicația client trecând în Starea 4.

În aceasta stare utilizatorului i se cere să introducă numele utilizator si parola, necesare pentru a se autentifica în sistem. Dacă clientul nu dispune de un cont şi o parolă, el nu poate trece la partea de realizare a lucrării de laborator.

După validarea autentificării utilizatorului, se trece în Starea 5, care reprezintă faza propriu-zisă de antrenare (training), adică efectuează lucrarea de laborator virtuală. Aceasta stare se repetă până ce studentul parcurge toți pașii lucrării. El nu poate parcurge laboratorul decât în ordinea proiectată, anumite butoane fiind inactive pentru a nu permite sărirea peste anumite operații sau navigare de la un modul la altul al lucrării fără realizarea celui precedent.

În cadrul antrenării utilizatorul va interacționa cu simulatorul prin intermediul zonelor definite pentru fiecare modul component al lucrării de laborator în parte prin care studentul introduce date şi primeşte rezultate sau imagini. Conținutul si forma datelor ce sunt trimise sunt verificate înainte de transmiterea acestora de către aplicația client pentru a se evita generarea unor erori care ar îngreuna comunicația client-server.

Când simularea lucrării ia sfârşit, aplicația client trece în Starea 6 în care utilizatorul îşi verifica cunoştințele acumulate în laboratorul pe care l-a parcurs. Această fază poartă numele de autoevaluare.

Aplicația client se termină cu Starea 7, unde utilizatorului i se afişează rezultatul obținut, întrebările la care a greşit precum şi explicațiile rezultatelor corecte. Eventual, el poate primi şi unele recomandări asupra căror module sa revină pentru o mai bună înțelegere a materialului de studiat.

În varianta pilot a sistemului LABVirt starile 6 si 7 nu au fost implementate, urmand ca acestea sa fie implementate într-o versiune ulterioară, am preferat inversarea starilor 3 cu 4 pentru o mai buna securitatea a sistemului si am ales ca incarcarea imaginilor sa se faca intr-o zona separata de cea de introducere a datelor si de obtinere a rezultatelor.

Descrierea aplicatiei server

Caracteristicile platformei gazda

Aplicatia server este proiectata sa ruleze pe sisteme de operare compatibile Windows NT. Alegerea platformei Windows are ca argument faptul ca exista un numar foarte mare de aplicatii de simulare cu suport pentru acesta platforma. Versiunile compatibile NT sunt de preferat datorita stabilitatii si a suportului imbunatatit pt. comunicatia în retea.

Separarea sesiunilor de lucru

Aplicatia trebuie sa suporte un numar de sesiuni de lucru client-simulator ce se vor executa în "paralel", drept pentru care devine necesara o separare a datelor si a codului pentru fiecare client în parte. Astfel, pentru orice conexiune, serverul va lansa pe un fir de executie codul ce asigura legatura client-simulator. Datele vor fi alocate pe o stiva proprie, integritatea lor fiind asigurata de mecanismele de protectie ale sistemului de operare.

Pentru cazul în care vor exista spatii de lucru comune, (mediul de lucru al simulatorului daca acesta este unic pentru mai multe sesiuni), fiecare fir de executie va modifica datele în asa fel incit acestea sa nu intre în conflict cu datele celorlalte fire.

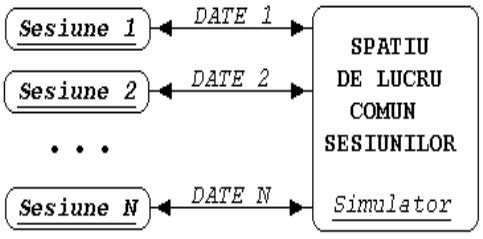

Fig.2. Sesiuni de lucru cu simulatorul

Avind la baza ideile susmentionate, se poate proiecta un singur modul, a carui instante vor reprezenta sesiunile si care se vor executate în "paralel", cu mentiunea ca, daca circumstantele o cer, datele ce vor fi schimbate cu

simulatorul vor fi independente de la o sesiune la alta.

Arhitectura aplicatiei server. Cooperarea cu modulul simulator

Înca de la inceput se disting doua parti functionale ale aplicatiei:

- un modul de initializare a structurilor aplicatiei si lansarea unei interfete necesara supravegherii si gestionarii activitatii pe server
- un modul ce va realiza conexiunea client – server – simulator. Acest modul se va lansa pe un fir de executie paralel cu cel precedent, la fiecare cerere de conectare venita din parte unei aplicatii client.

Aplicatia debuteaza cu un set de initializari realizate de firul principal. Sunt inițializate:

- un fir de executie pe care va rula socketul de ascultare a retelei pe portul 1500, port ales ad-hoc de catre echipa executive, din care am facut parte. Structura firului de server o voi prezenta ulterior.
- un alt fir de executie ce va monitoriza firele de executie corespunzatoare sesiunilor de lucru si le va realoca în cazul în care acestea au ajuns la un final.
- interfața cu mediul de simulare
- o interfata cu utilizatorul permitindu-i acestuia vizulizarea si gestiunea starii curente a serverului, urmind ca la comanda de iesire sa opreasca toate firele în executie, sa dezaloce datele alocate dinamic si sa termine aplicatia.

Firul de server

Firul de server are rolul de a asculta reteaua cu rolul depistarii cererilor de conectare venite din partea aplicatiilor client, cit si de a genera un fir de executie tip client-server-simulator pentru fiecare conexiune acceptata.

La primirea unei cereri de conectare, socketul firului server creează un nou socket, specific conexiunii în curs de acceptare, care este transmis firului client numai daca el îndeplineste condițiile de securitate prescrise de administratorul sistemului . Altfel, socketul copie este distrus si se reintră în bucla de asteptare a unei noi conexiuni.

Firul de server ruleaza conform schemei din figura 3:

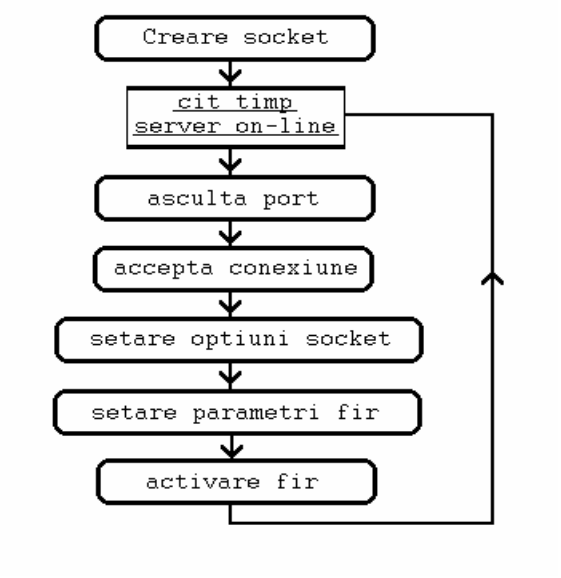

Fig.3. Execuție fir

Modulul alocator de fire

Datorita faptului ca un asemenea sistem nu poate sustine simultan un numar foarte mare de conexiuni client-server-simulator, s-a preferat o alocare statica a datelor, mai eficienta si mai putin pretentioasa decit o alocare dinamica. Pentru a gestiona optim conexiunile ce sunt pornite respectiv oprite s-a recomandat realizarea unui simplu alocator pentru firele de executie, care urmareste conexiunile eliberate si le dezaloca, respectiv aloca noile conexiuni în spatiile precedent eliberate si doar în cazul în care acestea nu exista este alocata o noua pozitie.

Modulul alocator este compus din doua parti:

- un fir ce ruleaza în paralel cu sistemul si elibereaza pozitiile firelor de executie terminate;
- o functie ce intoarce prima pozitie de la inceputul zonei alocate în care se poate porni un nou fir client-server-simulator. Această funcție este folosită de catre firul de server atunci cind aloca firele de client-server-simulator.

Modulul client-server-simulator

Acest modul va fi activat la fiecare cerere de conexiune acceptată de către server, având rolul de interfață între aplicația client şi aplicația de simulare. El are de realizat trei activități:

1. comunicația cu aplicația client, ce se va desfăşura pe baza protocolului de comunicație client-server
2. translatarea datelor între formatele simulator şi aplicația client
3. transmiterea lor către simulator, realizarea simulării şi extragerea datelor rezulatate.

Secventa de initializare a datelor firului

Se realizează următoarea secvență de inițializări (figura 4):

- Într-un prim pas este generat un identificator unic pentru firul în execuție, identificator pe baza căruia vor fi modificate şi identificate datele ce vor fi trimise spațiului comun al simulatorului, cu scopul precizat anterior, de a nu intra în conflict cu datele scrise de alte fire. Identificatorul este construit pe baza informațiilor legate de conexiunea TCP/IP şi este unic.
- Se alocă spații de lucru interne firului
- Se aloca variabile şi spații de lucru pentru interfața cu simulatorul
- Se aloca spații de lucru pentru fişierele de configurare.

Secvența de autentificare a utilizatorului

În protocolul de comunicație, după ce serverul trimte sirul de identificare, aplicația client trimite şi ea un sir de identificare, unic pentru un utilizator. Acest şir este interpretat şi cautat într-un fisier în care vor fi înregistrați utilizatorii autorizați ai aplicației. În cazul în care respectivul utilizator există, serverul raspunde cu "OK" şi se trece la secventa urmatoare. Altfel sesiunea va fi închisă şi firul terminat.

Secvența de creare şi comunicare a structurilor de date

Odata ce autentificarea s-a încheiat cu succes aplicația client selectează o lucrare de laborator prin comanda din protocol "LAB:xx", aplicația server realizează următoarele activități:

- încarca fişierele de configurare,
- îşi crează structurile de date funcție de conținutul lor
- le transmite şi aplicației client.

De la acest punct, comunicația cu sistemul la distanță va avea loc sincron, avînd la bază aceste structuri. Interpretarea fisierelor de configurare se va face cu ajutorul unor unor clase predestinate acestui scop.

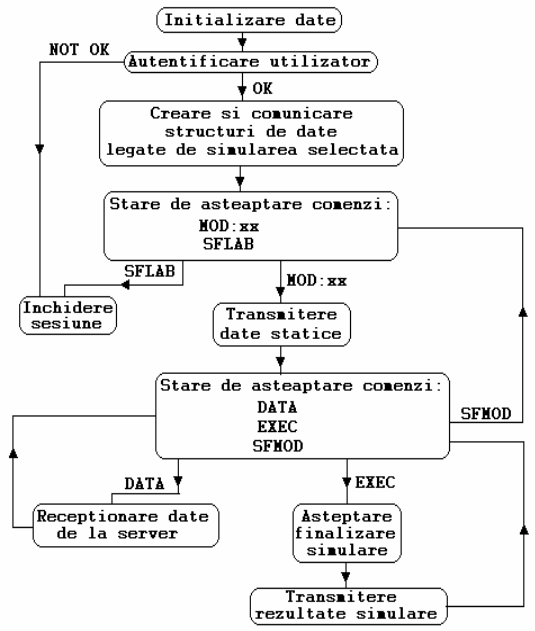

Fig.4. Starile firului client-server-simulator

Comenzile MOD:xx, SFLAB

Începând cu acest pas comunicația se va desfaşura sincron între client şi server, în sensul în care fiecare va parcurge structura proprie de date creată la pasul anterior şi va realiza transmisia de date în concordantă cu aceasta.

Comanda "MOD:xx" selectează modulul "xx" din structura lucrarii de laborator, modul ce va conține datele unei secvențe complete de simulare. Lucrarea de laborator poate fi alcătuită dintr-un număr suficient de mare de asemena module, ce vor fi parcurse consecutiv, iesirile simulărilor putînd reprezenta intrari pentru simulări imediat următoare.

O alta comanda care poate fi trimisă de catre client la acest pas este comanda "SFLAB", care permite o încheierea sesiunii, în urma ei ambele sisteme încheind conexiunea.

Secvența încheiere sesiune

La acest pas se ajunge în momentul în care serverul a primit comanda "SFLAB", urmatoarea actiune pe care o întreprinde fiind evident închiderea conexiunii, dezalocarea memoriei şi iesirea din firul de execuție.

Secvența de transmitere a instanțelor obiectelor

După ce serverul a primit comanda "MOD:xx", el transmite instanțele obiectelor statice descrise în fişierul de configurare .SCF către client, în ordinea în care acestea apar în fisierul de configurare (serverul este sincron cu aplicatia client din acest punct de vedere, deoarece ele procesează acelasi fisier de configurare). Datele trimise la acest pas vor fi atât textuale cât şi grafice, ele urmând a fi afişate utilizatorului, în cadrul aplicației client.

Comenzile DATA, EXEC, SFMOD

Datele parcurg traseul applet-server-simulator, în ambele sensuri, avand aceleasi valori dar reprezentări diferite(una la client si alta la simulator). De aceea este nevoie de un "translator" de format (figura 5). Tot aici, rezultă si necesitatea fisierelor de configurare, asa cum sunt descrise în sectiunea următoare. Pentru utilizarea fisierelor de configurare se utilizeaza un "interpretor" caracteristic fiecarui tip de fisier, care genereaza în memorie structuri de date.Structurile de date generate fac obiectul comunicatiei client-server-simulator (in ambele sensuri) ce este gestionata prin comenzile urmatoare:

Comanda "DATA" precede sirul instantelor obiectelor de intrare pentru simulare venite de la client. Serverul, după ce recepționează acestă listă, converteşte datele la formatul cunoscut simulatorului, şi se întoarce în bucla precedentă, asteptînd comanda de începere a simularii "EXEC".

Comanda "EXEC" este executata imediat după ce au fost primite datele de la client. În urma recepționării ei, serverul da comandă de început simularii, si intră într-o buclă de asteptate, ieşirea fiind condiționată de terminarea simularii. Cât timp acesta nu a ajuns la sfârşit, serverul are grija să transmită aplicatiei client un semnal de prezentă la fiecare 30 de secunde. Cind simularea s-a terminat serverul converteşte rezultatele în format interpretabil de catre aplicația client, şi le transmite acesteia tot în ordinea în care apar în fisierul de configurare.

Comanda "SFMOD" marchează sfârşitul unui modul, obligând serverul sa se decupleze de la modulul curent, şi sa revina în prima buclă de asteptare.

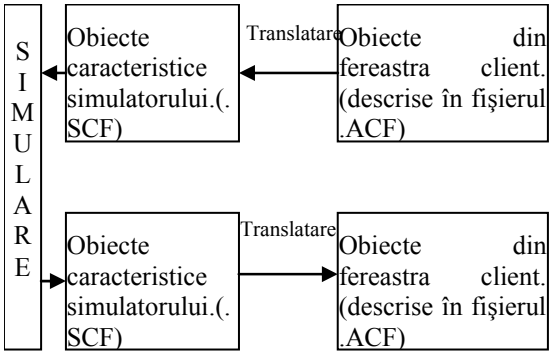

Fig. 5. Traseul client-server-simulator pentru un obiect

Fişiere de configurare: „.SCF", „.ACF", „.RES"

Fiecare lucrare de laborator va fi caracterizata prin următoarele trei tipuri de fişiere de configurare:

- „.ACF" (Applet Configuration File) fişierul ce conține configurația aplcației client;

Acest fişier este necesar configurării appletului pentru o anumită lucrare. Acest fisier este structurat pe mai multe module corespunzatoare fiecărei etape de simulare în parte. Un asemenea modul conține obiecte prin intermediul cărora se va interacționa cu utilizatorul. În definirea obiectelor se va avea în vedere un nume unic pentru fiecare obiect, coordonatele şi dimensiunile în fereastra utilizator, tipul de dată pe care îl va vizualiza (text/imagine), precum şi tipul de acces al utilizatorului la ea (citire/scriere).Un exemplu de obiect este descris mai jos:

NUME: TIP_DATE: TIP_TR: X0: Y0: DIM_X: DIM_Y

tip_date:T-text tip_tr:D-initializat

B-binar I-intrare

E-iesire

- „. SCF" (Server Configuration File) fişierul ce conține configurația aplicației server. Acest fişier are scopul de a realiza translatarea obiectelor din „.SCF" într-un format accesibil simulatorului şi invers. Fiecărui modul în parte îi este ataşat un nume de fişier ce va conține codul interpretabil al simulării.

- „.RES" (Resource File) fişierul ce conține resursele statice ale aplicației: texte şi imagini. Acestea vor exista pentru fiecare obiect descris în fişierul „.SCF" ca fiind de tipul „D" (inițializat), şi vor fi afişate în fereastra

aplicației client, pentru fiecare modul în parte.

Fişierele ce conţin codul interpretabil al simularii vor avea structura unei funcţii, în sensul că va avea un numar de date de intrare, un cod interpretabil, şi un număr de date de iesire.

Comunicația aplicație server–simulator utilizat

În general, pentru a fi apt să se integreze aplicației LABVirt, un simulator trebuie să aibă două proprietăți:

- să dispună de posibilitatea de a fi multiuser (mai multe sesiuni pot fi deschise simultan)
- să dispună de un mecanism de comunicare care să permită aplicației server să transfere date în mod eficient şi relativ comod şi să comande etapele de similare necesare.

Concluzii

În acest articol noi am încercat să arătam tehnologia folosită şi modalitatea de exploatare pe care noi o considerăm optimă pentru dezvoltarea sistemului de învățământ virtual şi mai ales a accesului la distanță a laboratoarelor. Mai sunt numeroase posibilități de dezvoltare a produsului software pe care noi le prezenta in materialele ce vor urma.

Bibliografie

[1] Nisioiu C.(2003) Realizarea aplicatiei client a sistemului LABVirt-proiect de diploma indrumat de Prof.univ.dr.ing. Minzu V.

[2] Silvestru C.(2003) Biblioteca digitala pentru informatica economica in cadrul conferintei nationale " Universitatea virtuala din Romania " – Academia de Studii Economice Bucuresti

[3] Silvestru C.(2005) E-learning in cadrul conferintei internationale de informatica economica " – Academia de Studii Economice Bucuresti.