UDK: 004.9 Stručni rad

APLIKACIJA ZA KONTROLU PRISTUPA RESURSIMA U RAČUNARSKIM LABORATORIJAMA

APPLICATION FOR RESOURCE ACCESS CONTROL IN COMPUTER LABORATORIES

Dušan Marković, Vukman Korać, Perica Štrbac
Visoka škola elektrotehnike i računartva strukovnih studija u Beogradu

dusanm.viser@gmail.com, vkorac@viser.edu.rs, pericas@viser.edu.rs

Rezime: *U ovom radu opisano je rešenje originalne aplikacije, bazirane na klijent-server arhitekturi i TCP (Transmission Control Protocol) komunikaciji, koja obavlja ulogu kontrole studentskog rada i studentskog pristupa resursima u računarskim laboratorijama. Aplikacija sa jednog (master) računara omogućuje uvođenje restrikcija u pristupu resursima na ostalim (slave) računarima u računarskoj mreži, odnosno, laboratoriji. Opisane su funkcionalnosti aplikacije i načini njihovih implementacija. Iz razloga kompatibilnosti sa različitim operativnim sistemima aplikacija je realizovana u Java(FX) tehnologiji. Prikazani su pripadni UML (Unified Modeling Language) dijagrami aktivnosti koji se odnose na klijentsku i serversku stranu rešenja. Aplikacija je implementirana i testirana na Windows operativnom sistemu u laboratorijama Visoke škole elektrotehnike i računarstva u Beogradu.*
Ključne reči: *Aplikacija, Java(FX), klijent-server, kontrola resursa, UML*

Abstract: *This paper describes the solution of the one original application based on client-server architecture and TCP (Transmission Control Protocol) communication, which performs the role of control of student's work and student's access to resources in the computer labs. The application from one (master) computer allows the introduction of restrictions in access to resources on other (slave) computers in a computer network, (laboratories). Describes the functionality of the application and ways of their implementation. For reasons of compatibility with different operating systems application is implemented in Java (FX) technology. Corresponding UML (Unified Modeling Language) activity diagrams relating to the client and server side solutions are shown. The application is implemented and tested on the Windows operating system in the laboratories of the School of Electrical and Computer Engineering of Applied Studies.*
Key words: *Application, Java(FX), client-server, resource control, UML*

1. UVOD

Na tržistu IT tehnologija postoje razna softverska rešenja koja omogućuju monitoring i kontrolu pristupa resursima računara u laboratorijama. Neke od takvih su iTalc (Intelligent Teaching And Learning with Computers) [1] i Epoptes [2] namenjeni za Windows odnosno Linux operativne sisteme. Uz pomoć ovih aplikacija, profesor – predavač može nadgledati rad studenata u računarskim laboratorijama kao i uvoditi restrikcije u pristupu određenim resursima a sve sa ciljem poboljšanja kvaliteta nastave kao i sprečavanja pokušaja varanja na ispitima ili kolokvijumima.

Glavni nedostatak pomenutih aplikacija je taj što ne omogućuju podešavanja svojstvena direktno vezanih za organizaciju lokalnih mrežnih resursa. U našem slučaju bilo je potrebno omugućiti kontrolu pristupa lokalnim mrežnim diskovima kao i lokalnim serverima.

Takođe je bilo potrebno naći rešenje koje će biti kompatibilno za izvršavanje na svim operativnim sistemima jer su računari, u zavisnosti od namene laboratorija, konfigurisani tako da rade na različitim operativnim sistemima (Windows, Linux, MAC OS). U tom smislu odabrana je tehnologija Java (FX) [3] kao rešenje kompatibilnosti.

2. FUNKCIONALNOSTI APLIKACIJE

Polaganje većine kolokvijuma i ispita na Visokoj elektrotehničkoj školi odvija se na računaru u jednoj od laboratorija tako da je bilo potrebno napraviti strategiju koja bi studentima maksimalno smanjila mogućnost prepisivanja, a time i povećala verodostojnost postignutih rezultata. Ideja je bila da se kreira centralizovani računarski sistem u kom bi centralni (master) računar imao funkcionalnost da kontroliše kojim resurima ostali računari mogu da pristupe.

Kontrola obuhvata dva dela kao što sledi:

- Blokiranje pristupa USB portovima čime studenti ne mogu da koriste materijale na svojim prenosnim memorijskim uređajima;
- Onemogućavanje kontrole nad mrežnim resursima kojima studenti mogu da pristupe.

U okviru onemogućavanja kontrole nad mrežnim resursima išlo se na onemogućavanje pristupa globalnoj računarskoj mreži (Internetu), a zatim i lokalnim mrežnim resursima kao sto su mrežni diskovi i Web server na kome se nalazi sajt Škole kao i Moodle LMS sistem. Na sajtu Škole i Moodle LMS sistemu se nalaze materijali predavanja i rešeni zadaci laboratoriskih vežbi. Kako bi se onemogućilo studentima da nastave sa radom i posle isteka vremena predviđenog za polaganje ispita ili kolokvijuma, u aplikaciju je implementirana funkcionalnost koja onemogućava rad posle isteka vremena testiranja.

Aplikacija je kreirana tako da restrikcije u pristupu mogu dobiti pojedinačni ili svi računari u laboratoriji. Takođe, moguće je uvesti i delimične restrikcije tako da je

moguće dopustiti parcijalan pristup resursima. Po završetku termina sa posebnim režimom rada, podešavanja se vraćaju na prvobitno stanje.

3. KLIJENT-SERVER ARHITEKTURA APLIKACIJE

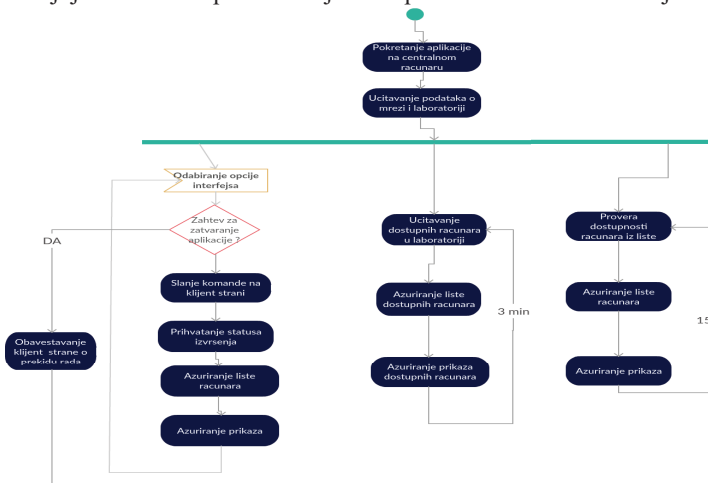
Aplikacija je realizovana kao klijent-server arhiktetura [4] bazirana na TCP komunikaciji [5] u računarskoj mreži [6]. Ova originalna aplikacija razlikuje dva tipa korisnika:

- Profesor (predavač)
- Student

Navedeni korisnici se se odnose na klijentsku u serversku stranu respektivno. Server predstavlja centralni računar na kome je instaliran deo aplikacije preko koje profesor (predavač) može da kontroliše rad na ostalim računarima. Pristup centralnom računaru kao i aplikaciji bi trebalo da imaju samo predavači kako ne bi došlo do zloupotrebe sistema. Na klijentskim računarima je instaliran drugi deo aplikacije koji je zadužen prvo da osluškuje da li je pokrenuta aplikacija na centralnom računaru, a zatim i da čeka komande, odnosno, zahteve za izvršenje koje dolaze sa tog računara. Nakon prijema komande, aplikacija izvršava sve potrebne izmene na klijentskom računaru a zatim vraća status izvršenja serverskom delu aplikacije.

3.1 SERVERSKI DEO APLIKACIJE

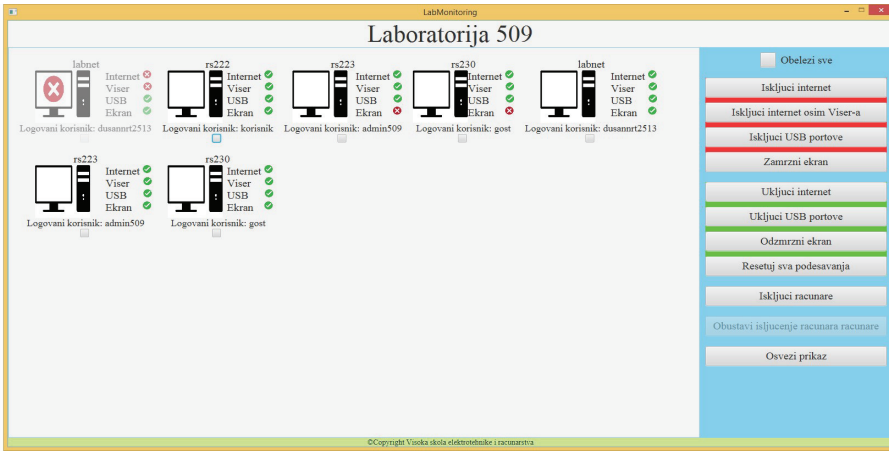
Na slici 1. prikazan je UML dijagram aktivosti [7] serverskog dela aplikacije. Nakon startovanja, aplikacija učitava sve informacije koje su vezane za mrežu u kojoj se nalazi računar kao i naziv laboratorije. Nakon toga pojavljuje se korisnički interfejs i pokreće mehanizam koji je zadužen za pretraživanje dostupnih računara u laboratoriji.



Slika 1. UML dijagram aktivnosti serverskog dela aplikacije

Postupak pretraživanja se ogleda u tome da se na svaku host adresu mreže šalje zahtev za otvaranje TCP sesije. Ukoliko se uspostavi sesija, server aplikacija šalje zahtev za registraciju računara i čeka se odgovor klijentske strane. Kada modul za pretraživanje dobije potvrdu, aplikacija dodaje taj računar u listu dostupnih klijenata i prikazuje ga u grafičkom interfejsu sa svim pristiglim podacima koji ga indentifikuju a zatim raskida sesiju. Ukoliko nakon određenog vremena ne dođe do otvaranja TCP sesije prelazi se na sledeću host adresu. Ceo proces pretraživanja se periodično ponavlja kako bi se obezbedilo realno stanje o dostupnim računarima.

Osim modula za pretraživanje, serverski deo aplikacije poseduje i modul koji ima za cilj da proverava da li su klijenti iz liste i dalje dostupni, te da li je moguća komunikacija sa njima. Ovaj modul je implementiran tako što salje UDP (datagramski) paket ka svakom klijentu iz liste i čeka da mu se paket u istom sadržaju vrati. Ukoliko nakon određenog vremena ne dobije odgovor postavlja atribut dostupnosti klijenta na trenutno nedostupan i osvežava prikaz u grafičkom interfejsu. U slučaju da se klijent ne odazove ni posle četvrte iteracije provere klijent se briše iz liste.



Slika 2. Korisnički interfejs serverskog dela aplikacije

Kako bi predavač imao vizuelni uvid u to koji su računari prisutni kao i koje su restrikcije primenjene na njima kreiran je korisnički interfejs koji se učitava pri pokretanju server aplikacije. Putem ovog korisničkog interfejsa predavač obavlja interakciju sa računarima na kojima se uvode ili ukidaju zabrane u pristupu.

Slika 2. prikazuje korisnički interfejs serverskog dela aplikacije gde je levo data grafička prezentacija klijentskih računara sa pripadnim atributima, dok je sa desne strane pozicioniran deo interfejsa koji serveru omugućuje slanje komandi za izvršavanje na klijentskim računarima. Na interfejsu su prikazani svi registrovani klijent računari iz liste sa svojim atributima koji ih indentifikuju.

Atributi o klijent računarima se odnose na identifikacione i restrikcione informacije vezane za taj računar (slika 2) kao što sledi:

- naziv radne stanice;
- korisničko ime (kojim je student prijavljen na dati računar);
- prava pristupa: Internetu, viseru (), USB-u i ekranu.

Postupak slanja serverskih komandi sastoji se iz dva koraka:

- Korisnik prvo bira računare kojima hoće da pošalje komandu;
- Zatim iz liste ponuđenih komandi bira onu koju želi za izvršenje.

Nakon što korisnik odabere komandu za izvršenje, serverski deo aplikacije se obraća izabranim klijent računarima putem UDP protokola u kome se navodi komanda koja se na njima treba izvršiti. Nakon slanja komande, klijent koji dobije i izvršava komandu obaveštava server stranu o statusu izvršenja komande. Kada server dobije potvrdan odgovor onda se u grafičkom prikazu menjaju stanja atributa koji definišu stanje zabrane pristupa datom resursu na datom klijentskom računaru. Ukoliko server ne dobije odgovor nakon određenog vremenskog perioda ili ako se u statusu obrade navede da je došlo do greške pri izvršavanju serverske komande na klijentskoj strani, korisnik dobija obaveštenje o grešci i u grafičkom prikazu klijenta neće doći do promene.

3.2 KLIJENTSKI DEO APLIKACIJE

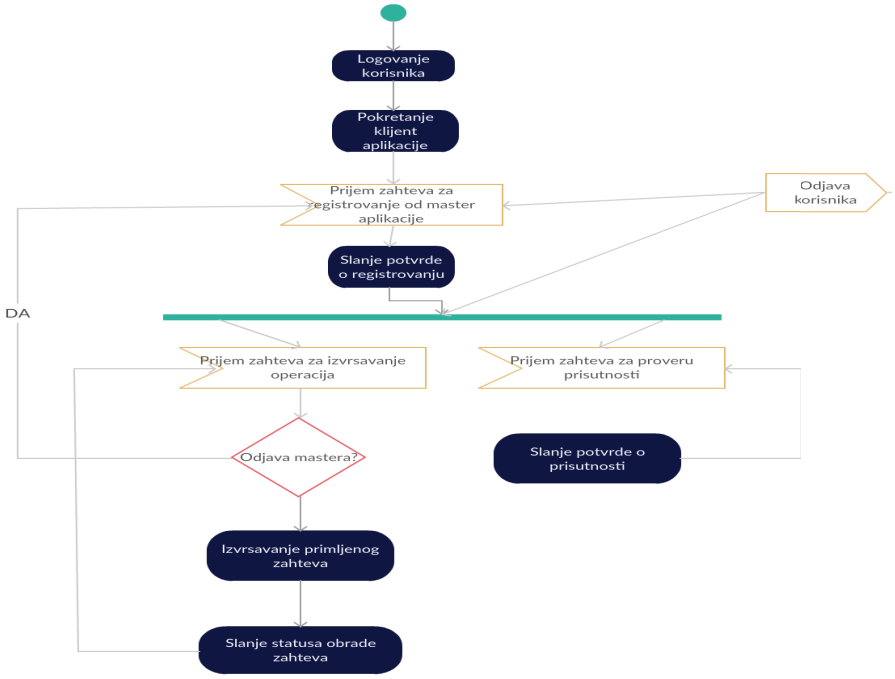
Na slici 3. dat je prikaz UML dijagrama aktivnosti klijentskog dela aplikacije prema napred navedenim opisima. Klijentski deo aplikacije instaliran je na svim računarima u laboratoriji osim centralnog računara i pokreće se prilikom prijavljivanja korisnika (studenta) na operativni sistem. Nakon pokretanja, aplikacija prikazuje obaveštenje studentu da je njegov rad moguće nadgledati i kontrolisati od strane predavača a zatim konfiguriše prava pristupa na podrazumevane vrednosti. Sama aplikacija ne poseduje grafički interfejs namenjen korisniku i izvršava se kao pozadinski proces. Zbog promena u konfiguraciji sistema kako bi se postigle tražene restrikcije, aplikacija se izvršava sa maksimalnim privilegijama.

Nakon što klijentski deo aplikacije izvrši početnu konfiguraciju na podrazumevane vrednosti ulazi se u stanje čekanja zahteva za registraciju od serverskog dela aplikacije. Po prijemu zahteva za registraciju od servera klijentski deo aplikacije vraća potvrdu u vidu indentfikacionih informacija koje opisuju klijenta. Time se obezbeđuje autentičnost u listi dostupnih računara na serverskoj strani. Moduli klijenstkog dela aplikacije koji se pokreću nakon registracije su:

- Modul za prijem i izvršavanje serverskih komandi;
- Modul za obaveštavanje serverske strane o dostupnosti klijenta.

Modul za prijem i izvršavanje serverskih komandi se pokreće posle izvršene registracije i njegova osnovna funkcija je da čeka zahtev od strane server aplikacije. Po prijemu serverske komande izvršava datu komandu a onda šalje status izvršenja te komande ka serveru. Ukoliko klijent aplikacija nije u mogućnosti da izvrši pristiglu operaciju vraća

negativan odgovor serveru kao i podatke o grešci pri izvršavanju serverske komande. Posle slanja statusa izvršenja, modul se vraća u stanje čekanja zahteva. U situaciji kada predavač pokrene postupak prekidanja rada server aplikacije, šalje se obaveštenje o prekidu rada svim registrovanim klijentima, koje prihvata modul za prijem i izvršavanje serverskih komandi, a zatim klijentska aplikacija odlazi u stanje čekanja zahteva za registraciju od strane servera pri čemu modul zadužen za obaveštavanje serverske strane o dostupnosti klijenta prestaje sa radom.



Slika 3. UML dijagram aktivnosti klijentskog dela aplikacije

Modul za obaveštavanje o dostupnosti čeka zahtev za proveru od serverskog dela i vraća pristigli sadržaj u istom formatu, čime se serveru daje potvrda da je klijent dostupan i spreman za prijem i izvršenje komandi servera. Rad ovog modula je direktno zavisan od rada modula za prijem i izvršavanje serverskih komandi.

4. Zaključak

U radu je prikazana originalna aplikacija koja omogućuje profesorima veću kontrolu i bolji uvid u rad studenata u računarskim laboratorijama u smislu sprečavanje zloupotrebe raspoloživih resursa prilikom rada studenata kao i postizanje verodostojnijih rezultata prilikom testiranja stečenog znanja. Aplikacija je implementirana korišćenjem Java(FX) tehnologije i realizovana je kao klijent-server arhitektura. Serverska strana ima mogućnost nadgledanja i postavljanja stanja klijentskog računara koja se odnose na pristup Internetu, lokalnim mrežnim resursima kao sto su mrežni diskovi i Web server na

kome se nalazi sajt Škole kao i Moodle LMS sistem, te USB-u i ekranu. Dat je opis funkcionalnosti serverske i klijenske strane aplikacije. Dati su pripadni UML dijagrami aktivnosti klijentske i serverske strane. Aplikacija je testirana u laboratorijama Visoke škole elektrotehnike i računarstva u Beogradu.

LITERATURA

[1] Doerffel, T. (2008). *iTALC User Manual*, GNU Tobias Doerffel, V1.0.4

[2] Georgopoulos, A. (2016). *Epoptes documentation*, http://www.epoptes.org, V0.5.10

[3] Johan, V., Weiqi, G., Stephen, C., Dean, I., James, W. (2014). *Pro Java Fx 8*, APRESS

[4] Haroon, S. O. (2014). *Client-Server Model*, IOSR Journal of Computer Engineering, Volume 16, Issue 1, Ver. IX, pp 67-71

[5] Zhao, L. (2012). *Using UDP Datagram to Realize a Distributed Control Mode at High-Speed Data Communication*, Elsevier, Physics Procedia 25, pp. 886 – 891

[6] Harold, E. R. (2014). *Java Network Programming*, O'Reilly, 4th edition.

[7] Andre, E., Choppy, C., Reggio, G. (2014). *Activity Diagrams Patterns for Modeling Business Processes*, Software Engineering Research, Management and Applications, pp. 197-213.