



Vilniaus Universitetas

Pavyzdinis banko serverio modelis

*Kompiuterių tinklų pratybų užduotis.
Projekto dokumentacija*

Autorius:

**Ignas Butėnas
VU MIF, Informatika
II kursas, 9 grupė
2007-03-15**

Autorius:

Modelį kūrė ir realizavo Vilniaus Universiteto Matematikos ir Informatikos Fakulteto, II kurso, 9 grupės studentas Ignas Butėnas. Kontaktinė informacija: ignas.butenas@mif.vu.lt.

Darbo tikslai:

Pagrindinis darbo tikslas buvo susipažinti su tinklo programavimu. Užduotis – sukurti serverį ir su juo bendraujantį klientą tam panaudojant Windows Sockets. Šiam tikslui įgyvendinti buvo pasirinktas projektas „Pavyzdinis banko serverio modelis“. Buvo išsikelti tikslai naudoti objektinį programavimą. Vienintelis tikslas, kuris nebuvo iškeltas tai – praktinis panaudojimas. Praktiniam panaudojimui visų pirma reiktų daug efektyvesnio serverio (visų pirma turėtų veikti gijos (threads)), antra reiktų užlopyti saugumo spragas ir aprašyti algoritmus leidžiančius apsaugoti duomenų bazę ir trečia tai pareikalautų daug daugiau laiko nei jo yra skirta užduočiai atlikti. Taigi projekto rezultatas – pavyzdinis banko serveris, veikiantis tinkle ir demonstruojantis kaip būtų galima viską plėtoti toliau.

Darbo aprašymas:

SERVERIS

Po paleidimo serveris paklausia kokį portą naudoti kliento priėmimui ir tada laukia kol klientas prie jo prisijungs. Prisijungus atliekami tolimesni veiksmai. Serveris vienu metu gali apdoroti tik vieną varotoją. Kiti laukia eilėje ir tik vienam pabaigus darbą atsiranda vieta kitam.

Galimybės:

Serveris gali priimti vartotojo užklausas ir jas vykdyti. Gavęs užklausą serveris tikrina ją su savo komandų aibe ir radęs tinkamą atlieka veiksmus. Rezultatai grąžinami eilute, kuri vėliau paruošiama siuntimui ir išsiunčiama klientui suprantamu būdu.

Galimi veiksmai:

- prisijungimas prie duomenų bazės (toliau - DB) (komanda: **login**);
- atsijungimas nuo DB (komanda: **logout**);
- einamojo vartotojo vardas (komanda: **user**);
- sąskaitos papildymas (komanda: **put**);
- pinigų nuėmimas iš sąskaitos (komanda: **get**);
- sąskaitos einamasis likutis (komanda: **balance**);
- pagalba (komanda: **help**);

Naudojimasis:

Paleisti serverį yra labai paprasta – tereikia paleisti **server.exe** failą. Paleistas serveris paklaus kokį portą jam naudoti, o po to jau sulaukus kliento vyks bendravimas su juo.

Naudojami failai (duomenų bazė):

Serveris duomenims laikyti naudoja failą **data.txt** esantį toje pačioje direktorijoje kaip ir **server.exe**. Norėdami sukurti naują vartotoją sistemoje tiesiog į failą įrašykite naują eilutę

„Vardas suma“

Kur vardas nurodomas raidėmis, o suma – sveikuoju skaičiumi.

KLIENTAS

Po paleidimo klientas bando jungtis prie serverio per tamtikrą portą ir į tam tikrą hostą. Šie nustatymai nurodomi konfigūracijos faile „**config.txt**“. **Dėmesio:** programos testavimui hostą palikite „localhost“, o portas turi sutapti su tuo portu kuriuo paleisite veikti serverį!

Galimybės:

Klientas gali bendrauti su serveriu užklausomis ir gauti atsakymus iš jo. Kliente nėra realizuota jokių specialių funkcijų, tai tiesiog užklausų siuntėjas serveriui. Gavęs atgalinę žinutę jis ją atvaizduoja konsolėje.

Naudojimasis:

Kliento panaudojimas labai paprastas. Žinodami prie kokio porto Jums reikia jungtis, nurodykite jį konfigūracijos faile. Pagal nutylėjimą ten nurodytas 1234 portas. Jei prisijungimas – sėkmingas iš serverio gausite pasveikinimo žinutę. Tada jau galite pradėti bendrauti su serveriu. Visas serverio komandas sužinosite surinkę **help** ir nuspaudę **ENTER** (taip užklausa siunčiama serveriui). **Pastaba:** baigę darbą surinkite komandą **exit**, kad serveris ilgiau nebelauktų Jūsų užklausų ir leistų dirbti kitam vartotojui. Būkite mandagūs kitų atžvilgiu.

Naudojami failai:

Vienintelis naudojamas failas yra **config.txt**. Čia saugomi kliento prisijungimo nustatymai.

Juos pakeisti galite tiesiog pakeisdami ten esamų duomenų reikšmes.

Pastaba: programos testavimui hostas būtinai turi būti „localhost“.

Problemos ir sprendimai:

Didžiausia problema buvo serverio projektavimas. Buvo būtina numatyti kaip jis elgsis gavęs vieną ar kitą užklausą. Kai projektas buvo sukurtas, teko imtis programavimo darbų. Serveriui ir klientui rašyti buvo pasirinkta C++ kalba ir objektinis programavimas. Objektinis programavimas pasirinktas todėl, kad programuojant taip kodas tampa daug lengviau skaitomas, efektyvesnis. Bet kaip visada būna pasirinkus tokį stilių prarandama atminties resursų ir greičio, tačiau šiais laikais tie nuostoliai mažai ką reiškia ir šio projekto atžvilgiu jie yra beveik nepastebimi. Programuojant buvo nuspręsta visą modelio atliekamą darbą pavesti serverinei pusei, kad klientui liktų tik atvaizduoti tai ką jis gauna iš serverio. Dar viena problema, kuri autoriui ilgai nedavė ramybės – buvo įvairių duomenų tipų konvertavimas vienas į kitą. Kadangi klientui buvo galima siųsti tik simbolių masyvą, tai viską serveris turėjo ir pateikti kaip simbolių masyvą. Taigi skaičiams ir kitiems duomenims konvertuoti į eilutę, o paskui į simbolių masyvus buvo apsirašytos funkcijos leidžiančios lengvai manipuluoti tipais. Tam panaudota standartinės C ir C++ kalbos bibliotekos, bei eilučių srautai (string streams).

Žinomos, bet neišspręstos problemos:

Didžiausia problema trukdanti naudoti tokį modelį praktikoje yra jo vienu metu apdorojamų klientų skaičius – 1. Žinomas ir problemos sprendimas (gijos), tačiau jis nėra realizuotas šiame modelyje. Dėl vieno kliento apdorojimo serveris turi polinkį „lūžti“ jei tik klientas atsijungs netvarkingai. „Lūžimas“ pasireškia tuo, kad serveris tiesiog nebelaukia kitų klientų, nes nežino ar pirmasis baigė darbą ar ne. Dalinis sprendimas – tvarkingas atsijungimas (*komanda exit*). Jei klientas tvarkingai atsijungia, kitas esantis eilėje gali be trikdžių dirbti su serveriu.

Išvados:

Pavyzdinis modelis veikia, o tai reiškia, kad sugalvotas projektas buvo įgyvendintas. Kodas yra lengvai skaitomas ir redaguojamas. Įterpti naują funkciją į serverį ir nesugriauti jo veikimo yra labai paprasta – tereikia aprašyti naują f-ją kuri gražintų eilutę. Tai įgalina greitai plėsti serverio galimybes. Objektinio stiliaus panaudojimas leidžia serveriui lengvai bendrauti su vartotojo informacija, duomenų baze. Be visa ko šio stiliaus pasirinkimas suteikia galimybę su minimaliomis pastangomis serveriui uždėti grafinį interfeisą.
