Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Študijný program: Počítačové systémy a siete

Tímový projekt

Simulátor komunikácie v počítačovej sieti

Tím č.1:
Bc. Adam Hamšík
Bc. Marián Schmotzer
Bc. Tomáš Mózes
Bc. Peter Péti
Bc. Maroš Nemsila

Garant predmetu: Ing. Ján Hudec
Vedúci projektu: Ing. Igor Grellneth PhD.

---

20. mája 2008

Obsah

Zoznam obrázkov

Kapitola 0

Úvod

0.1 Účel a rozsah dokumentu

Predkladaný dokument obsahuje analýzu, špecikáciu, návrh a implementáciu softvérového systému určeného na simuláciu počítačovej siete. Dokument je výsledkom dvojsemestrálneho študentského projektu v predmete "Tímový projekt" študijného odboru Počítačové systémy a siete na Fakulte informatiky a informačných technológií Slovenskej technickej univerzity v Bratislave. Dokument je určený pre študentov, pedagógov fakulty a pre posúdenie práce riešiteľmi z Tímu č.8 v rovnakom predmete.

0.2 Prehľad dokumentu

Dokument je členený do kapitol. Prvá kapitola obsahuje všeobecnú analýzu a úvod do počítačových sietí pre čitateľa, ktorý s touto oblasťou náhodou nie je oboznámený. Ďalej nasleduje prehľad existujúcich riešení simulácie a emulácie počítačových sietí. Druhá kapitola obsahuje špecikáciu požiadaviek na systém. Tretia kapitola ponúka návrh riešenia požiadaviek z predošlej kapitoly. Štvrtá kapitola nadväzuje na návrh a obsahuje popis prototypu aplikácie. Popisuje jednotlivé komponenty aplikácie a ich vzájomné prepojenie. V záverečnej piatej kapitole sú obsiahnuté implementačné detaily finálneho produktu. Popisujú sa proces spustenia emulátora, zmeny v databáze, webovské prostredie a rovnako aj proxy.

0.3 Zadanie úlohy

Navrhnite a zrealizujte programový systém pre simuláciu sieťovej komunikácie na druhej a tretej vrstve sieťovej architektúry RM OSI.

Systém má umožňovať:

- definovanie topológie simulovanej siete
- simuláciu rôznych prepájacích zariadení (napr. prepínač, smerovač, firewall ...)
- simuláciu komunikácie medzi prepájacími zariadeniami.

Funkčnosť navrhnutého systému overte v sieti so simulovanými zariadeniami pomocou komunikácie medzi koncovými zariadeniami.

0.4 Skratky

ASIC – Application Specific Integrated Circuit

CSMA/CD – Carrier Sense Multiple Access with Collision Detection

DRAM – Dynamic Random Access Memory

EEPROM – Electrically Erasable Programmable Read Only Memory

EIGRP – Enhanced Interior Gateway Routing Protocol

FTP – File Transfer Protocol

IEEE – Institute for Electrical and Electronics Engineers

IETF – Internet Engineering Task Force

ICMP – Internet Control Message Protocol

IOS – Internetworking Operating System

ISO – International Organization for Standardization

IP – Internet Protocol

JIT – Just In Time

LAN – Local Area Network

MAC – Media Access Control

MAN – Metropolitan Area Network

NVRAM – Non Volatile Random Access Memory

OSI – Open System Interconnection (basic reference model)

OSPF – Open Shortest Path First

PAN – Personal Area Network

PDU – Protocol Data Unit

RAM – Random Access Memory

RFC – Request For Comment

RIP – Router Information Protocol

ROM – Read Only Memory

SRAM – Static Random Access Memory

SSH – Secure SHell

STP – Spanning Tree Protocol

TACACS – Terminal Access Controller Access Control System

TCP – Transmission Control Protocol

TCP/IP – Transmission Control Protocol/Internet Protocol

UDP – User Datagram Protocol

VLAN – Virtual Local Area Network

WAN – Wide Area Network

Kapitola 1

Analýza

1.1 Analýza problematiky

1.1.1 Úvod do počítačových sietí

Počítačová sieť je vzájomné prepojenie viacerých počítačov. Prepojením sa sledujú viaceré ciele:

- zdieľanie informácií
- zdieľanie prostriedkov
- zvýšenie spoľahlivosti

Počítačové siete sa najčastejšie delia podľa rozlohy na PAN - personálne siete, LAN - lokálne siete, MAN - mestské siete, WAN - rozľahlé siete.

PAN Sú to rozlohou najmenšie siete. Siahajú na vzdialenosť niekoľkých metrov, viac menej na dosah ruky. Preto sa aj nazývajú personálne.

LAN Lokálne siete sa väčšinou vzťahujú k budove alebo komplexu budov a tak nepresahujú veľkosť niekoľko kilometrov. Vzhľadom na malé vzdialenosti dosahujú tieto siete najväčšie rýchlosti.

MAN Metropolitné siete rozsahom pokrývajú mestá a slúžia na prepojenie susedných LAN sietí.

WAN Rozľahlé siete prepájajú vzdialené LAN siete. Sú rozsahovo neobmedzené, môžu pokrývať alebo spájať celé kontinenty.

Model

Pre uľahčenie učenia, odhaľovania chýb, vývoja a interoperability sú siete popísané vrstvovým modelom, ktorý rozdeľuje komunikáciu v sieti do jednotlivých vrstiev. Každá vrstva poskytuje svoje služby vrstve nad ňou a zároveň využíva služby nižšej vrstvy. Pri komunikácii medzi uzlami siete dochádza k výmene dát v podobe protokolových dátových jednotiek (PDU - protocol data unit) medzi rovnakými vrstvami v daných uzloch.

Dva najviac používané modely sú OSI (open systems interconnection) model od organizácie ISO a TCP/IP (transmission control protocol/internet protocol) model organizácie IETF používaný pre popis komunikácie v Internete.

OSI model

OSI model siete sa skladá zo siedmych vrstiev (obr. 3.14).

Obr. 1.1: Referenčný sieťový model RM-OSI

1. **Fyzická vrstva** - Najnižšia vrstva nemôže využívať služby daľšej nižšej vrstvy, preto musí sama zabezpečovať fyzické spojenie najbližších uzlov siete, identifikáciu dátových okruhov, radenie bitov alebo skupín bitov. Na prenos informácií sa využívajú konkrétne médiá, ktoré sú však definované mimo OSI modelu. Model definuje iba rozhrania k týmto médiám.

2. **Linková vrstva** – Jej úlohou je zabezpečiť komunikáciu fyzicky prepojených uzlov. Definuje topológiu siete a spôsob adresácie zariadení na spoločnom médiu.

3. **Sieťová vrstva** – Zabezpečuje prepojenie vzájomne komunikujúcich koncových uzlov siete.

4. **Transportná vrstva** – Realizuje prenos dát medzi aplikáciami. Umožňuje komunikáciu viacerých aplikácií medzi počítačmi.

5. **Relačná vrstva** – Vytvára relácie medzi komunikujúcimi aplikáciami, pričom riadi vzájomný dialóg týchto aplikácií.

6. **Prezentačná vrstva** – Transformáciou a formátovaním dát zabezpečuje ich transparentný prenos medzi aplikáciami a ich jednotnú reprezentáciu bez ohľadu na použité kódovanie.

7. **Aplikačná vrstva** – Poskytuje svoje služby priamo používateľským aplikáciám. Ponúka spôsob vzájomnej komunikácie pre aplikácie bežiace na rôznych uzloch siete. Zabezpečuje dohodu o použitých službách nižších vrstiev (kódovanie, štruktúra, oprava chýb)

TCP/IP model

Aj napriek tomu že je OSI model štandardizovaný a využíva sa na výuku sietí, v praxi sa takmer nepoužíva. Tu je rozšírenejší druhý model - TCP/IP, pretože priamo popisuje architektúru Internetu tak, ako je implementovaný. Model TCP/IP je jednoduchší, obsahuje menej vrstiev (obr. 1.2), ktoré v sebe zahrňujú funkcie jednotlivých vrstiev OSI modelu

Obr. 1.2: Porovnanie referenčných modelov OSI (vľavo) a TCP/IP (vpravo)

1. **Vrstva sieťového prístupu** - Vrstva závislá od použitej sieťovej technológie. Zabezpečuje, aby mohol uzol komunikovať s typom siete, do ktorej je fyzicky pripojený. Najznámejšími protokolmi tejto vrstvy je Ethernet a Wifi v LAN sieťach a mnohé protokoly WAN sietí, ako napríklad ISDN, Frame Relay, ATM, DSL a podobne.

2. **Internetová vrstva** - Poskytuje nezaručené prepojenie koncových uzlov tak ako je to definované IP (Internet Protocol) protokolom. Taktiež vykonáva funkcie oznamovania chýb prostredníctvom ICMP protokolu.

3. **Transportná vrstva** - Transportná vrstva je tu rovnaká ako v OSI modeli. Definuje dva protokoly: TCP a UDP. TCP poskytuje zaručené prepojenie aplikácií vytvorením virtuálneho spojenia, v ktorom garantuje správnosť dát aj ich správne poradie. Doručenie správnych dát je zabezpečené vyžiadaním retransmisie v prípade prijatia poškodeného datagramu.

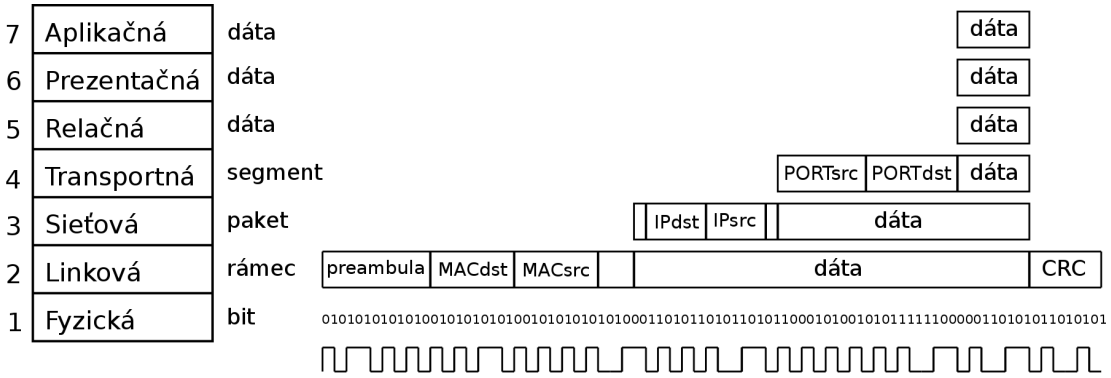
4. **Aplikačná vrstva** - Zahrňuje v sebe funkcie aplikačnej, prezentačnej a relačnej vrstvy OSI modelu. Patria sem dva typy protokolov: používateľské a podporné. Najbežnejšie používateľské protokoly sú SMTP, FTP alebo Telnet. Medzi podporné protokoly patria DNS, DHCP a SNMP. Tieto neponúkajú služby priamo používateľom, zabezpečujú však štandardné funkcie aplikácií.

Pre jednoduchšiu orientáciu čitateľa však aj pri ďalšom vysvetľovaní budeme používať OSI model, ktorý je pri výučbe rozšírenejší.

Komunikácia v sieti

Vysielanie Aplikácie pre vzájomnú výmenu dát využívajú služby nižších vrstiev. Dáta sú pri tom predávané nižším vrstvám a pri prijatí zase vyšším vrstvám tak, ako vidieť na obr. 1.3. Dáta z aplikácie rozdeľuje transportná vrstva do segmentov a pridáva k nim informáciu identifikujúcu zdrojovú a cieľovú aplikáciu. Aplikácie sú v počítači identifikované číslom portu (PORTsrc,PORTdst). Následne sú segmenty predané nižšej, sieťovej vrstve, kde sú segmenty enkapsulované do paketov, obsahujúcich okrem iného aj IP adresu zdrojového a cieľového počítača (IPsrc, IPdst). Pakety sú predané linkovej vrstve, ktorá zabalí paket do rámca, obsahujúceho spravidla zdrojovú a cieľovú fyzickú adresu (MACsrc,MACdst) a pridá kontrolnú

sumu (CRC). Rámec odovzdá fyzickej vrstve, ktorá sa postará o jeho transformáciu do podoby vhodnej pre vyslanie po konkrétnom type média použitého v sieti.



Obr. 1.3: Enkapsulácia dát pred vysielaním

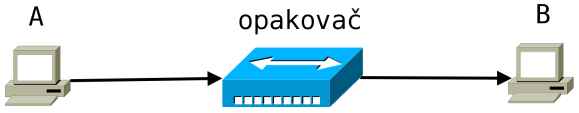
Príjem Fyzická vrstva premieňa signály z média na dáta, z ktorých linková vrstva vyberá rámce. Ak je rámec určený pre daný uzol, vyberie sa z neho paket a predá ho internetovej vrstve. Tá na základe adresy rozhodne kam sa ma paket ďalej poslať, prípadne ak je určený danej stanici, predá ho transportnej vrstve. Transportná vrstva podľa čísla portu vie, ktorej aplikácii v operačnom systéme sú dáta určené a tak jej po kontrole tieto dáta odovzdá.

1.1.2 Sieťové prvky

V tejto časti popíšeme sieťové prvky na všetkých vrstvách ISO OSI modelu. Začneme od prvkov, ktoré pracujú na prvej vrstve.

Opakovač (Repeater)

Opakovač je jedným z najjednoduchších sieťových prvkov. Jedná sa o dvojportové zariadenie, ktorého úlohou je regenerovať prijatý signál. Používa sa najmä kvôli útlmu signálu vo vodiči. Každý vodič po ktorom prechádza elektrický signál má určitý odpor a čím vzdialenejšie sú od seba komunikujúce strany, tým je signál slabší. Ak je signál príliš slabý, cieľový uzol ho nemusí vôbec zaregistrovať a tým pádom môže prísť ku strate údajov. Opakovač pracuje na prvej vrstve, čiže s vysielanými údajmi nevykonáva žiadnu zmenu ani filtráciu. Jeho úlohou je len prijať signál na jednom porte, zregenerovať ho do pôvodného stavu a vyslať ho na druhý port. V dnešnej dobe sa už prakticky nepoužíva, nakoľko je len dvojportový a nevykonáva žiadnu filtráciu.

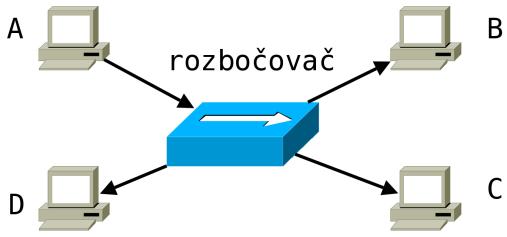


Obr. 1.4: Opakovač

Na obrázku č.1.4 je viditeľná komunikácia medzi uzlami A, B. Ak uzol A vyšle informáciu na segment, kde sa nachádza opakovač, ten jednoducho tento signál prepne sa svoj druhý port. Nevykonáva sa žiadna filtrácia, opakovač tu slúži len na predĺženie segmentu.

Rozbočovač (Hub)

Vylepšením opakovača je rozbočovač, ktorý je viacportový. Niekedy sa označuje ako viacportový opakovač. Jeho funkcionalita je totožná s jeho predchodcom, čiže signál prijatý na jednom porte sa vysiela na všetky porty okrem toho, na ktorom bol prijatý. Rozbočovače rozdeľujeme na aktívne a pasívne. Pasívne sú jednoduchšie, pretože prijatý signál neregenerujú, ale len ho preposielajú na výstupné porty. Aktívne rozbočovače sa od pasívnych líšia v tom, že vstupný signál regenerujú do pôvodného stavu. Ak je teda dĺžka segmentu veľmi dlhá, môže nám aktívny rozbočovač zabrániť, aby bol na cieľovom uzle ignorovaný. Tieto zariadenia sa taktiež v súčasnosti nepoužívajú z bezpečnostného hľadiska. Problémom je skutočnosť, že rozbočovač prijatú správu vysiela na všetky svoje výstupné porty. Ak teda vysielajúci uzol používa protokol FTP a nadväzuje so serverom spojenie, všetky zariadenia pripojené na rozbočovač túto komunikáciu taktiež uvidia. Problémom sú citlivé údaje ako heslá a podobne. A práve FTP alebo TELNET posielajú heslá nešifrované, čiže prístupové údaje môže zistiť každý, kto je k zariadeniu pripojený. Ďalším problémom týchto prvkov je, že pripojené zariadenia môžu komunikovať iba v režime Half-Duplex, nakoľko rozbočovač predstavuje jednu kolíznu aj broadcastovú doménu. Pri Half-Duplex móde môže zariadenie len vysielať alebo len prijímať údaje, nikdy nie zároveň obe.



Obr. 1.5: Rozbočovač

V tejto jednoduchej topológii (na obr.1.5) máme zapojené štyri počítače. V tomto prípade počítač A vysiela údaje uzlu D. Keď sa informácia dostane na rozbočovač, ten jednoducho signál zregeneruje a pošle na všetky svoje porty. Samozrejme, že údaje sú prijaté aj uzlami B a C. Toto je neefektívne, nakoľko počítače musia informáciu spracovať a jednak to predstavuje aj bezpečnostné riziko, pretože tieto počítače môžu odpočúvať komunikáciu.

Most (Bridge)

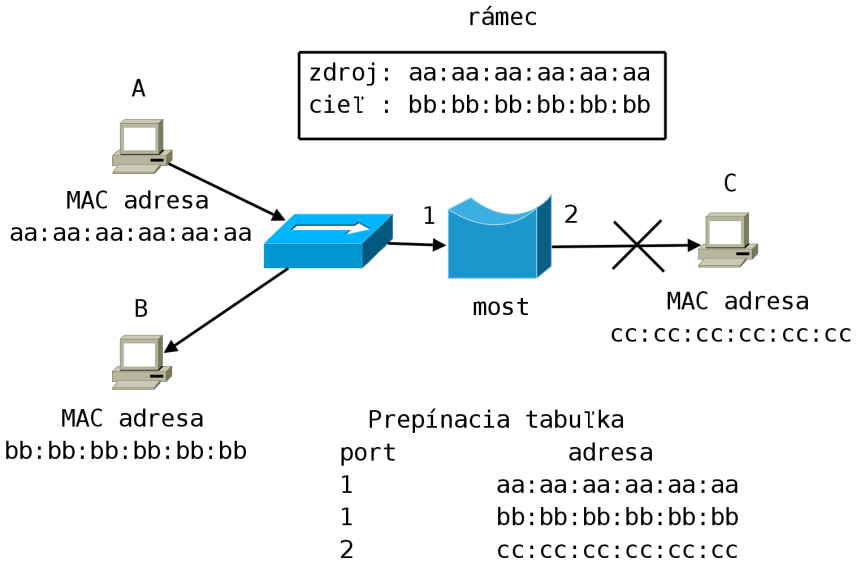
Most predstavuje analógiu s opakovačom. Takisto je dvojportový a prepína komunikáciu z jedného rozhrania na druhé. Problémom opakovača je skutočnosť, že neposkytuje filtrovanie, teda komunikáciu preposiela aj na druhé rozhranie zakaždým. Most práve slúži na oddelenie dvoch segmentov a teda segmentuje sieť. Ak do topológie vložíme most, dostaneme dve oddelené kolízne domény. Broadcastová doména je však jediná.

Most je inteligentnejšie zariadenie, pretože pracuje na druhej vrstve OSI modelu. Kým opakovače a rozbočovače pracovali s údajmi len vo formáte signálov, most používa prepína-

ciu tabuľku (switching table). Do tejto tabuľky si ukladá informácie o všetkých pripojených uzloch, ktorú potom použije na rozhodnutie, či prijatý rámec vyšle na druhé rozhranie. Do tabuľky je vždy uložená dvojica adresa-port. Most sa o sieťových uzloch učí prostredníctvom zdrojových adries.

Po prijatí rámca sa prečíta cieľová MAC adresa a tá sa vyhľadá v prepínacej tabuľke. V prípade ak je cieľová adresa na rovnakom porte na ktorom prišla správa, tento rámec je zahodený a nie je preposlaný na druhé rozhranie. Takýmto spôsobom sa vykonáva filtrovanie, čiže zabraňuje sa šíreniu rámcov do segmentov, kde to nie je nutné. Ak sa cieľová MAC adresa v tabuľke nenachádza, alebo sa táto adresa nachádza na druhom porte, most tento rámec prepošle na opačné rozhranie. Napokon sa most pozrie na zdrojovú MAC adresu rámca a skontroluje, či ju má uloženú vo svojej tabuľke. Ak ešte v tabuľke nie je, je do nej pridaná.

Nevýhodou mostu je jeho pomalšia softvérová implementácia, výhodou je oddelenie kolíznych domén a filtrácia na druhej vrstve OSI modelu. Toto filtrovanie nám zároveň poskytuje aj základný stupeň ochrany pripojených uzlov.



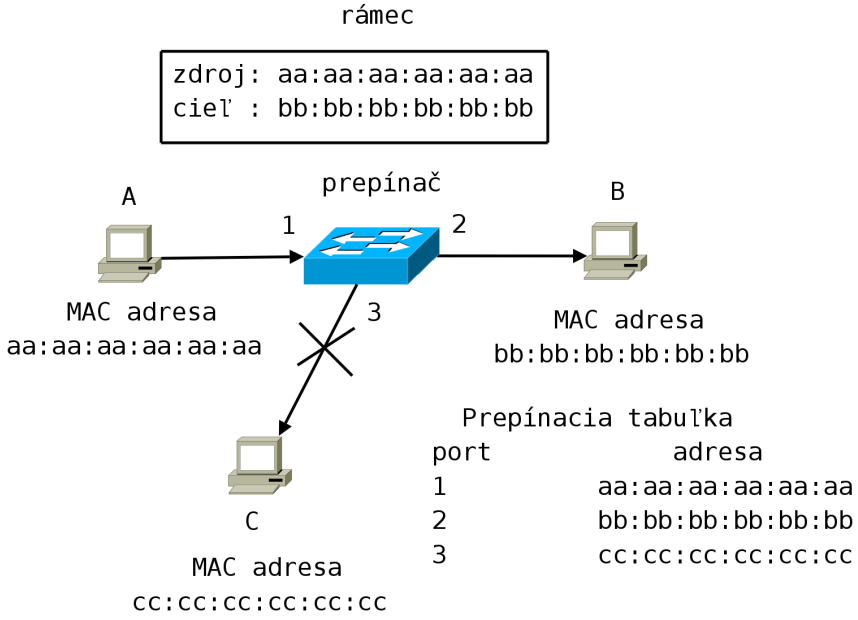
Obr. 1.6: Topológia s použitím mosta

Na topológii (obr. 1.6) je ilustrovaná základná funkcionalita mosta. V tomto príklade už predpokladáme, že sa most dynamicky naučil MAC adresy všetkých uzlov. Ak uzol A vyšle rámec s destinačnou adresou uzla B, tak po prijatí ho most zahodí, pretože zistil, že oba uzly sú na jednom segmente. Rámec sa teda dostane len na uzol B, uzol C ho vôbec nedostane. Most v tomto prípade vykonal filtráciu.

Prepínač (Switch)

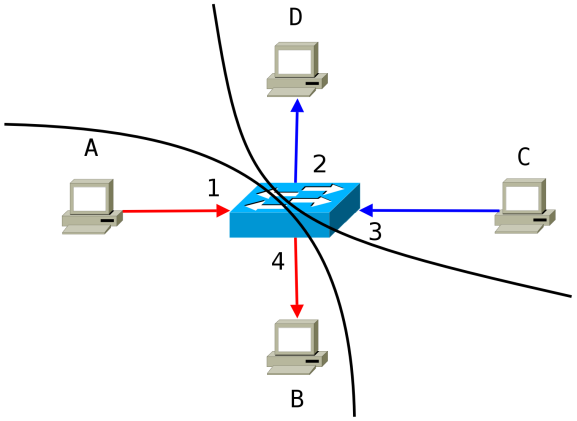
Prepínač vznikol zdokonalením mostu na viacportové zariadenie. V podstate je to rovnaký kontrast ako opakovač a rozbočovač. Prepínač je najpoužívanejším zariadením pracujúcim na druhej vrstve OSI. Princíp funkcie prepínača je rovnaký ako pri moste, rozdielom však je jeho rýchlejšia hardvérová implementácia pomocou ASIC. Ďalšou jeho vlastnosťou je mikrosegmentácia siete, ktorá umožňuje niekoľko paralelných prenosov. Kým zariadenia pripojené

do rozbočovača mohli pracovať len v režime Half-Duplex, pri prepínači je umožnená práca vo Full-Duplex, čiže údaje môžu byť zo zariadenia vysielané aj prijímané súčasne. Pri komunikácií dvoch pripojených zariadení sa na prepínači vytvára virtuálny okruh, ktorý je platný po celú dobu komunikácie. Vplyvom mikrosegmentácie sa každé priamo pripojené zariadenie nachádza vo vlastnej kolíznej doméne.



Obr. 1.7: Topológia s prepínačom

Prepínač na obrázku 1.7 mikrosegmentuje sieť a rámec z uzla A do B vysiela len na port 2. Počítač C o tejto komunikácii nič nevie.

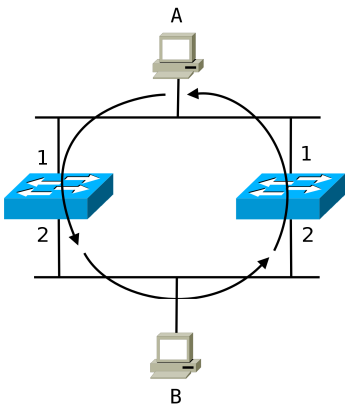


Obr. 1.8: Mikrosegmentácia siete pomocou prepínača

Nech sú k prepínaču pripojené uzly A, B, C, D (obr. 1.8). Ak spolu komunikujú zariadenia A-B a C-D, tento prenos je na prepínači realizovaný súčasne, čiže v jednom okamžiku

môžu vysielať a prijímať údaje všetky zariadenia. Pri tejto komunikácii sa prejavia výhody mikrosegmentácie siete, pretože jednotlivé uzly môžu využívať maximálnu kapacitu linky. Toto je obrovská výhoda oproti rozbočovaču, ktorý žiadnym spôsobom nesegmentuje sieť a nedovoľuje žiadne paralelné komunikácie. Pri použití rozbočovača bolo možné vykonávať len jednosmernú komunikáciu a navyše len half-duplex. Prepínač oproti nemu prináša bezpečnosť, veľkú rýchlosť a redundanciu.

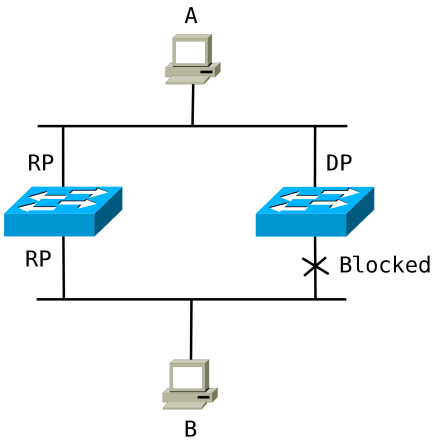
Spanning Tree Protocol Práve spomenutá redundancia predstavuje spôsob ako zabezpečiť vysokú dostupnosť siete. Redundancia v tomto ponímaní predstavuje vytváranie duplicitných spojení, ktoré môžu spôsobiť vytvorenie fyzických slučiek v počítačovej sieti. Fyzické slučky ale predstavujú problém, pri ktorom nám niektoré údaje môžu krúžiť dookola. Prepínače na predchádzanie tejto situácie používajú protokoly na vytvorenie bezslučkových topológií z akejkoľvek fyzickej topológie. Táto technológia sa volá Spanning Tree Protocol (STP) a je definovaná štandardom IEEE 802.1D. Po zavedení virtuálnych sietí sa začali používať aj modifikácie tohto protokolu ako Per-VLAN STP, Multiple STP alebo Rapid STP.



Obr. 1.9: Vznik slučiek v topológii s redundanciou

Uvažujme topológiu na obr. 1.9. Medzi oboma segmentami sú kvôli redundancii zapojené dva prepínače a nepoužívame STP. Keď počítač A vyšle broadcast, oba prepínače ho príjmu a do svojej prepínacej tabuľky si poznačia, že na porte 1 sa nachádza počítač A. Takto prijatý rámec vysielajú na spodný segment, pretože broadcastové správy sa preposielajú na všetky rozhrania okrem toho, na ktorom bol rámec prijatý. Tento rámec je prijatý uzlom B, ale takisto aj oboma prepínačmi. Vo svojej tabuľke si upravia lokáciu uzla A na port 2 a opäť vyšlú rámec na vrchný segment. Tento kolobeh bude prebiehať neustále. Týmto sa jednak zahltí kapacita prepínača, ale takisto aj počítačov.

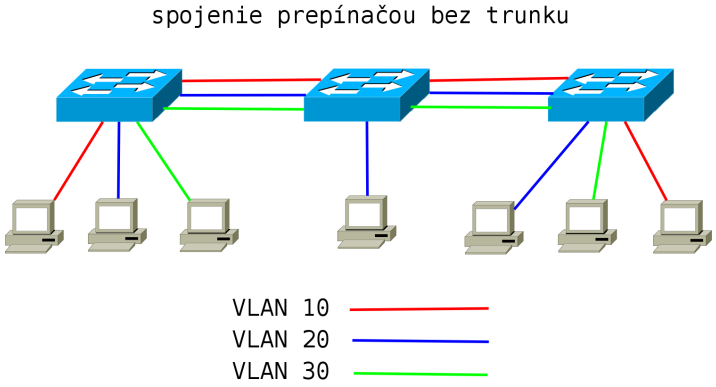
Práve kvôli odstráneniu tohto problému a zachovaniu redundancie sa používa STP, ktorý redundantné cesty zablokuje. Prepínače si periodicky každé 2 sekundy vymieňajú informácie v podobe BPDU (Bridge Protocol Data Unit), čím zistia topológiu siete. Pri tomto procese si vyberú koreňový (root) uzol a vytvoria najkratšiu cestu od každého prepínača ku koreňovému, a tak vytvoria graf. Porty prepínačov, ktoré nepatria do kostry grafu budú zablokované. Blokované porty ignorujú dátové rámce, ale BPDU si vymieňajú rovnako kvôli možnej zmene topológie. Predchádzajúci príklad bude pri zapnutí STP vyzerať ako je zobrazené na obr. 1.10.



Obr. 1.10: Princíp činnosti "spanning tree" protokolu

Prepínače si vybrali ľavý prepínač ako koreňový a jeho porty sú označené ako RP (Root port). Tieto porty vedú od koreňového prepínača k ostatným. Na druhom prepínači je jeden port označený ako DP (Designated port) a predstavuje cestu ku koreňovému prepínaču. Druhý port je zablokovaný (Blocked), čiže neprenáša žiadne údaje, ale prijíma BPDU. Touto metódou sú z topológie odstránené fyzické slučky vytvorením logickej topológie.

VLAN Ďalšou kľúčovou technológiu je možnosť vytvárania tzv. virtuálnych sietí (VLAN). Pomocou VLAN dokážeme na jednom prepínači definovať rôzne broadcastové domény, pričom na komunikáciu medzi nimi použijeme zariadenie pracujúce na tretej vrstve OSI modelu - smerovač.

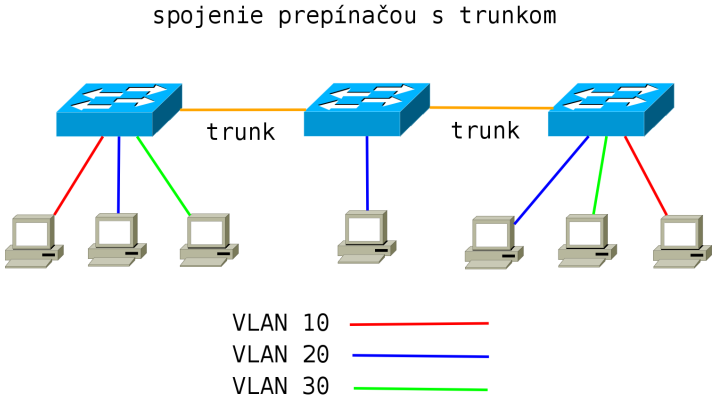


Obr. 1.11: Prepojenie prepínačov s viacerými VLAN

Na obrázku máme prepojené tri prepínače, pričom sme definovali tri virtuálne siete, nazvané sú VLAN 10, VLAN 20 a VLAN 30. Klasicky sa pri prepojení prepínačov použije pre každú VLAN jeden port.

Kvôli zníženiu nákladov na výstavbu VLAN a na zamedzeniu plytvaniu portov prepínačov sa používa metóda trunk podľa štandardu IEEE 802.1Q. Pomocou trunku dokážeme na

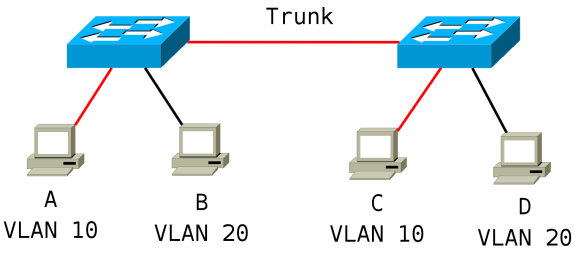
jednom fyzickom prepojení prenášať údaje z niekoľkých VLAN. Predchádzajúcu topológiu zjednodušíme tak ako vidieť na obr. 1.12.



Obr. 1.12: Prepojenie prepínačov s použitím "trunk" technológie

Multiplexovaním VLAN cez jeden fyzický spoj sme ušetrili osem portov. Pri tomto zapojení je dôležité si uvedomiť, že porty definované ako trunk nepatria do žiadnej VLAN, prenášajú totiž všetky VLAN. Pri použití trunku musíme samozrejme zabezpečiť, aby prepínače jednoznačne vedeli, do ktorej VLAN jednotlivé rámce patria. Toto môžeme zabezpečiť IEEE štandardom 802.1Q (nazývané tiež dot1q), alebo proprietárnym Cisco protokolom ISL (Inter-switch Link). Kým 802.1Q rozširuje Ethernet hlavičku o dodatočné dva bajty a umožní zadefinovať až 4094 VLAN, ISL pôvodný rámec zapúzdri a pridá novú 26-bajtovú hlavičku. 802.1Q sa používa častejšie, pretože zaberá len dodatočné dva bajty a je otvoreným štandardom. Cisco ISL dokonca nepodporujú ani všetky Cisco zariadenia.

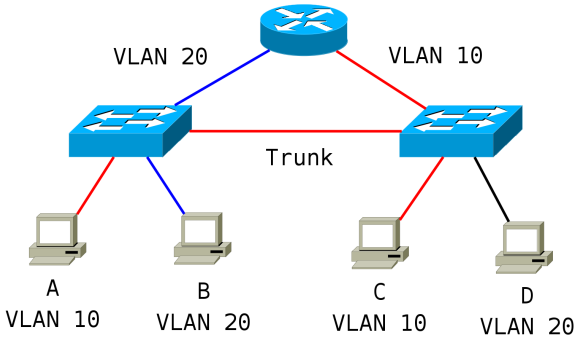
Princíp komunikácie v rámci VLAN si môžeme ilustrovať na príklade (obr. 1.13):



Obr. 1.13: Komunikácia v rámci VLAN

Na obrázku 1.13 máme prepojené dva prepínače, pričom na každý sme zároveň pripojili dva počítače. Porty prepínača patria do príslušných virtuálnych sietí. Predpokladajme, že si prepínače ešte nestihli vytvoriť prepínaciu tabuľku a príkazom ping chceme overiť konektivitu medzi uzlami A a C. Počítač A musí najprv vyslať ARP request na získanie fyzickej adresy počítača. Keď táto požiadavka dorazí na prepínač, ten ho smie preposlať iba do rovnakej VLAN, kam patrí prichádzajúca správa. Správa sa teda môže poslať len na druhý prepínač, ktorý správu pošle len na port, kde sa nachádza počítač C. Virtuálne siete nám v podstate rozdelia jeden fyzický prepínač na viacero logických, pričom ako tradične sa na komunikáciu

medzi rôznymi sieťami použije zariadenie pracujúce na 3. vrstve - smerovač. Komunikácia medzi uzlami C a B bude prebiehať podľa obr. 1.14:

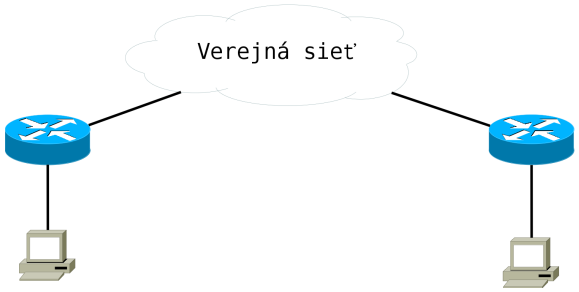


Obr. 1.14: Prepojenie VLAN pomocou smerovača

Uzol C vyšle správu, tú v rámci VLAN 10 prepínač pošle na smerovač a takisto na druhý prepínač. Na druhej strane sa doručí na počítač A, pretože je v rovnakej VLAN, ale táto správa bude ignorovaná. Dôležité je, že smerovač dokáže paket smerovať a vyslať ho do VLAN 20, čím sa dostane až k počítaču B.

Smerovač (Router)

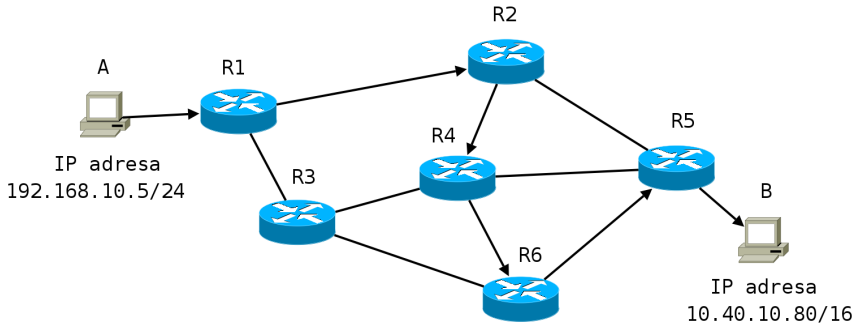
Jedným z najmocnejších zariadení v počítačovej sieti je smerovač. Ako jediné zo spomenutých zariadení pracuje na tretej vrstve OSI modelu. Kým prepínač na druhej vrstve pracuje s MAC adresami, smerovač pracuje s IP adresami. MAC adresy sú dané priamo od výrobcov sieťových kariet ku ktorým sa viažu a nie sú teda vhodné na smerovanie v rámci Internetu. IP adresy sú na druhej strane hierarchické, čo umožňuje, aby sme mohli správu doručiť ľubovoľnému uzlu.



Obr. 1.15: Prepojenie sietí smerovačmi

Ako je zrejmé z obrázka 1.15, hlavnou úlohou smerovača je preniesť správu z jedného miesta na iné. Vo všeobecnosti smerovače umožňujú komunikáciu medzi ľubovoľnými dvoma bodmi. Táto komunikácia sa väčšinou uskutočňuje vo verejnej sieti, akou je celosvetový Internet.

Smerovače používajú smerovacie tabuľky, pomocou ktorých sa rozhodujú kam sa jednotlivé pakety budú odosielať. V tejto tabuľke sú významné najmä dve hodnoty - sieť a ďalší skok (next hop). Smerovač pracuje softvérovo, takže ak prijme paket na svojom rozhraní, prechádza postupne všetky položky v smerovacej tabuľke až kým nenájde zhodu. Ak smerovač vie kade paket poslať, využije next hop tejto destinácie. Cesty do neznámych sietí sú podľa nastavenia buď zahadzované, alebo sa posielajú na predom definovaný port.



Obr. 1.16: Hľadanie cesty smerovačmi

Statické smerovanie Jedným z problémov, ktoré musia smerovače riešiť je nájdenie najlepšej cesty do vzdialenej siete. Existujú dva spôsoby ako sa smerovače učia o vzdialených sieťach, tzv. staticky a dynamicky. Sieťový administrátor zadáva pri statickom smerovaní destinácie priamo. Takýto prístup poskytuje vysokú rýchlosť a bezpečnosť, pretože po sieti sa neposielajú žiadne informačné pakety. Bezpečnosť môžeme chápať aj v tom zmysle, že administrátor každej destinácii predpisuje presnú cestu. Na druhej strane je takýto prístup nevhodný pre veľké topológie, ktoré obsahujú viacero smerovačov a množstvo pripojených sietí. Takéto riešenie je možné, ale je zdĺhavé a náchylné na chyby. Najväčšou nevýhodou je statickosť ciest, čiže pri zlyhaní predpísanej cesty sa nevyhľadá alternatívna.

Dynamické smerovanie Druhý, dynamický prístup je založený na učení sa ciest od ďalších smerovačov. Smerovače si v tomto prípade medzi sebou podľa určitého algoritmu vymieňajú informácie pomocou ktorých sa učia o nových cestách. Týchto algoritmov je tiež niekoľko a opäť sa dajú rozdeliť na dve skupiny - Distance Vector (DV) a Link State (LS). Dôležitou charakteristikou všetkých spomenutých algoritmov sú kritéria, na základe ktorých vyberajú najlepšie cesty - metriky.

Distance–vector protokoly Pri distance vector protokoloch si susedné smerovače periodicky vymieňajú kompletné smerovacie tabuľky. V rámci protokolu je definovaný vektor vzdialenosti, ktorý každému cieľovému uzlu priradí vzdialenosť. Smerovače si pomocou Bellman-Ford algoritmu vypočítajú najkratšiu cestu ku každému uzlu. Kritérium výberu najlepšej cesty sa nazýva metrika. Metrikou môže byť počet smerovačov medzi zdrojom a cieľom, prenosová kapacita linky, oneskorenie a podobne. Nevýhodou DV je ich pomalá konvergencia a náchylnosť na smerovacie slučky. Na riešenie problémom zacyklenia sa používajú techniky ako split horizon, route poisoning alebo triggered updates. Aj použitím týchto metód môžu v

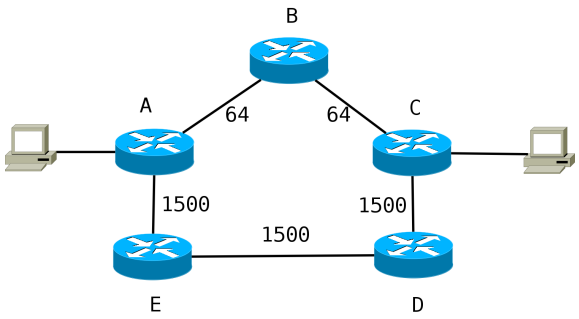
určitých prípadoch v sieti vzniknúť smerovacie cykly. DV sú veľmi rozšírené a často využívané najmä kvôli svojej jednoduchosti.

Najvýznamnejšími predstaviteľmi DV sú RIP verzie 1 a 2, IGRP a EIGRP. RIP v1 (Routing information protocol) je najstarším DV algoritmom. Jeho metrikou je počet skokov medzi smerovačmi a interval vysielania smerovacích tabuliek je 30 sekúnd. Problémom je spôsob vysielania aktualizácií, nakoľko používajú broadcast. RIP v2 je vylepšením pôvodného RIP v1, pričom podporuje VLSM (Variable length subnet masks), autentifikáciou a aktualizácie sa vysielajú multicastom. RIP je otvorený štandard a navzdory chybám je vďaka svojej jednoduchosti konfigurácie veľmi obľúbený.

IGRP (Interiour gateway routing protocol) je proprietárny protokol firmy Cisco. Je to pokročilejší smerovací protokol, pretože využíva kompozitnú metriku. Na výpočet najlepších ciest nepoužíva počet skokov, ale prenosovú kapacitu linky a oneskorenie na sieti. Voliteľne môže použiť aj ďalšie parametre.

EIGRP (Enhanced IGRP) sa vyhlasuje za hybridný smerovací protokol, ale v skutočnosti predstavuje DVA. Vylepšením oproti predchádzajúcim protokolom je ich rýchlejšia konvergencia, nakoľko si udržiava aj alternatívne cesty do sietí. Za hybridný sa vyhlasuje aj z toho dôvodu, že udržiava tabuľku susedov.

Dôležitosť metriky je viditeľná na obr. 1.17:



Obr. 1.17: Metriky pre spoje medzi smerovačmi

V tejto oblasti máme zapojených 5 smerovačov, pričom linky medzi nimi sú rýchlostí 64 a 1500 kbps. Cieľom je dostať sa zo smerovača A do C. RIP používa ako metriku počet skokov (hop-count), čiže podľa neho je najvýhodnejšou cestou postupnosť A, B, C. Ako si môžeme všimnúť, cesta A, E, D, C má rýchlosti niekoľkonásobne vyššie. Všetky protokoly, ktoré majú priepustnosť linky (bandwidth) ako jednu z metrík vyberú rýchlejšiu cestu.

Link state protokoly Link state algoritmy pracujú na odlišnom princípe ako DVA. Smerovače si pri link state protokoloch nevymieňajú smerovacie tabuľky, ale informácie o pripojených linkách. Každý smerovač má teda rovnaký prehľad o počítačovej sieti. Pomocou Dijkstrovho algorimu si následne každý smerovač vypočítava najkratšiu cestu do všetkých sietí autonómne. Pri DVA si smerovače vymieňali už hotové smerovacie tabuľky, pri LSA si smerovače vymieňajú informácie o stave svojich liniek a až následne si z týchto údajov vypočítavajú svoju unikátnu smerovaciu tabuľku. Výhodou tohto prístupu je vysoká rýchlosť konvergencie a nízke zaťaženie siete. Nevýhodou je väčšia náročnosť na hardvér smerovača, nakoľko sa všetky cesty prepočítavajú po každej zmene v sieti. Riešením tohto problému

je definovanie samostatných oblastí, pričom lokálne významové informácie sa neohlasujú do susedných oblastí.

Predstaviteľom LSA je OSPF (Open shortest path first). OSPF využíva Dijstrov algoritmus na hľadanie najkratších ciest v sieti a ako metriku používa prenosovú kapacitu linku. Jeho veľkou výhodou je rýchla konvergencia a možnosť rozdeliť sieť na oblasti. Aj keď je zjavne výhodnejším ako iné protokoly, je náročnejší na konfiguráciu aj hardvér smerovačov.

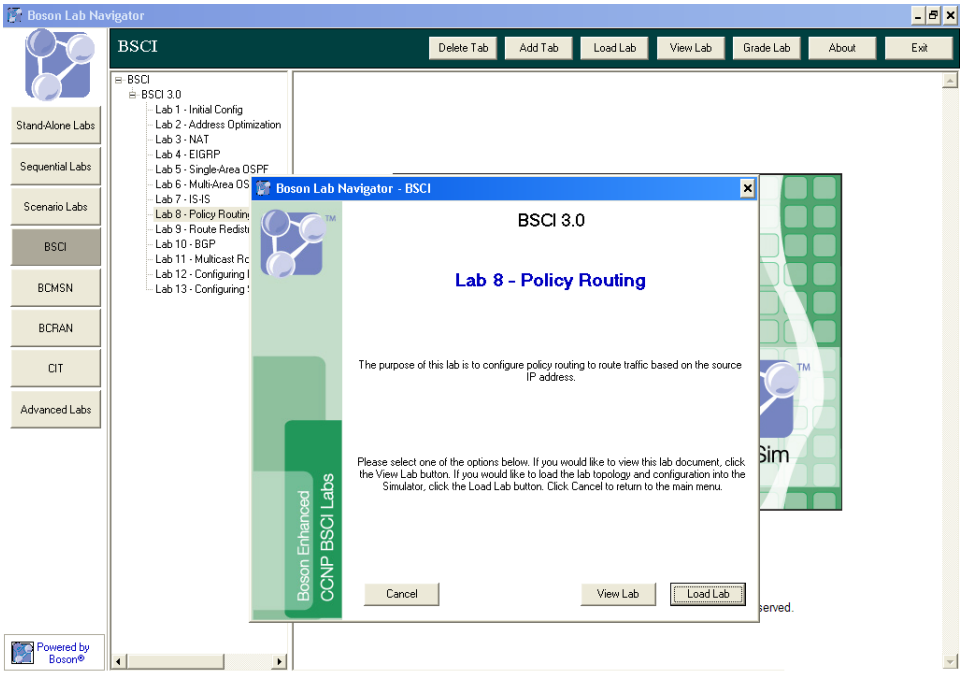
1.2 Analýza existujúcich riešení

Táto časť sa zameriava na analýzu už existujúcich riešní, ktoré sa v súčasnosti používajú na simuláciu, alebo emuláciu siete. Aplikácií, ktoré umožňujú túto funkcionalitu je viacero. Tu sa zameriame iba na najrozšírenejšie a najkomplexnejšie riešenia.

1.2.1 Boson NetSim

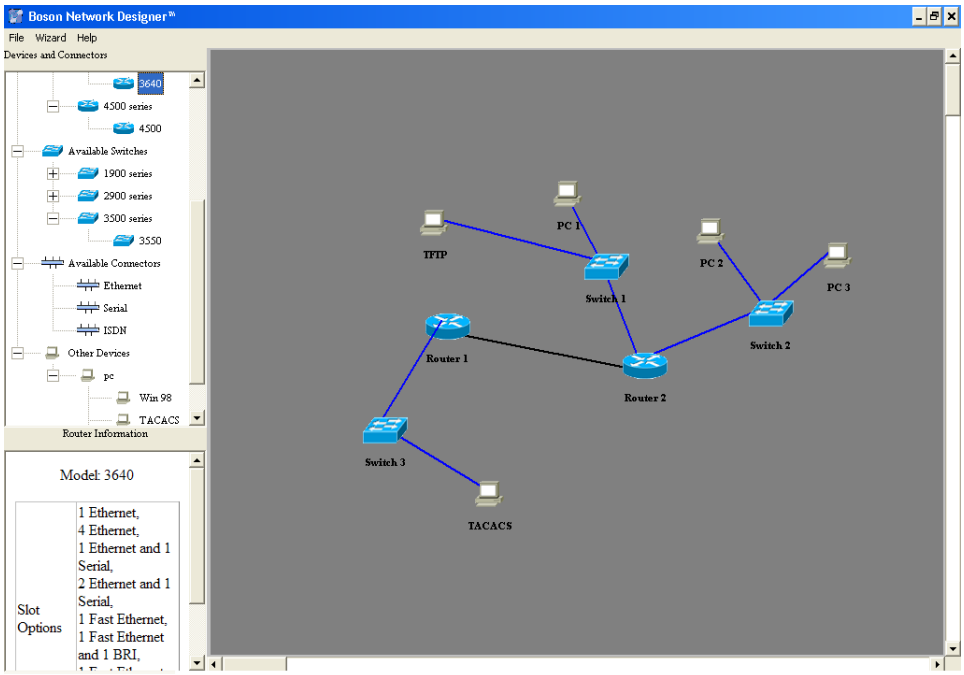
Prvou aplikáciou je Boson Netsim (www.boson.com). Jedná sa o simulátor, ktorý pozostáva z troch častí.

Boson Lab Navigator – je štrukturovaný zoznam vytvorených konfigurácií. Na precvičenie rôznych funkcií sú tu návody, ktoré stručne popisujú jednotlivé príkazy od úplneho začiatku. Ku každému návodu je vytvorené zapojenie zariadení, ktoré je možné rovno spustiť v simulátore.



Obr. 1.18: Ukážka simulátora Boson Lab Navigator

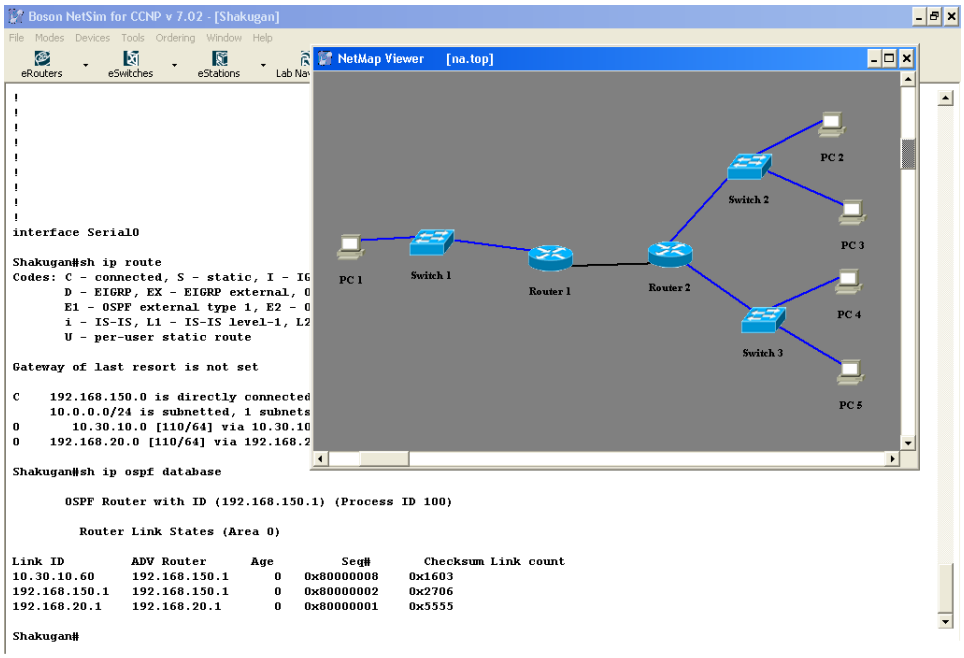
Boson Network Designer – umožňuje vytvárať vlastné sieťové zapojenia. Každá konfigurácia môže obsahovať cez 200 zariadení. Pri navrhovaní vlastného zapojenia máme k dispozícii 47 rozdielnych zariadení. Ponúkané série smerovačov sú : 800, 1000, 1600, 1700, 2500, 2600, 3600 a 4500. Z každej série je možné si vybrať niektorý z modelov, ktoré sa odlišujú sieťovými rozhraniami. Pri niektorých modeloch sa používateľovi ponúka možnosť špecifikovať počet portov a typ sieťových kariet. Pre simuláciu prepínačov sú dostupné tieto modely : 1912, 2950 a 3550. Do konfigurácie je možné zapojiť aj simulovaný počítač (príkazový riadok Windows 98), autentifikačný server TACACS, alebo TFTP server. V porovnaní s ostatnými aplikáciami je prostredie pre modelovanie zapojenia na slabšej úrovni, niektoré nastavenia nie sú spravené veľmi prehľadne (napr. konfigurácia Frame Relay zapojenia).



Obr. 1.19: Ukážka simulátora Boson Network Designer

Boson NetSim – je aplikácia vykonávajúca samotnú simuláciu. Po načítaní zapojenia ponúka zoznam vybraných zariadení a možnosť ich konfigurovať. Pripojenie na CLI konzolu je možné realizovať v okne tejto aplikácie, alebo telnet-ovým pripojením v samostatnom okne. Prostredie simulátora umožňuje vytvorenie väčšiny zapojení, s ktorými sa môžete stretnúť pri štúdiu CCNA a CCNP. Medzi nevýhody simulátora patrí obmedzenie, že bez opätovného načítania zapojenia ho už nie je možné meniť, čím sa vynuluje konfigurácia zariadení. Taktiež sa počas konfigurácie možete stretnúť s chybami v CLI, napr. pri výpise parametrov k príkazom. Implementácia obsahuje aj zopár iných drobných chýb prostredia.

Hlavné okno simulátora nám ponúka možnosť načítať vytvorenú topológiu. Jednotlivé prvky siete môžeme konfigurovať v grafickom prostredí programu pomocou konzoly. Na obrázku 1.20 vidíme aj topológiu siete a samostatné okno pre jednu inštanciu smerovača.



Obr. 1.20: Ukážka simulátora Boson Netsim

Veľkou výhodou simulátora je možnosť riešiť niektorú z prednastavených sieťových topológii. Pomocou nástroja Boson Lab Navigator si môžeme vybrať sieť podľa našich znalostí, pričom niektoré zapojenia sú určené pre začiatočníkov CCNA a niektoré pre skúsenejších CCNP.

Aplikácia obsahuje drobné chyby a niektoré riešenia nie sú veľmi priateľské voči používateľovi. Medzi nevýhody môžeme zaradiť aj fakt, že sa jedná iba o simuláciu zariadení, preto sa pri skutočných zariadeniach môžeme stretnúť s určitými rozdielmi či už pri konfigurovaní, alebo funkcionalite.

1.2.2 Packet Tracer

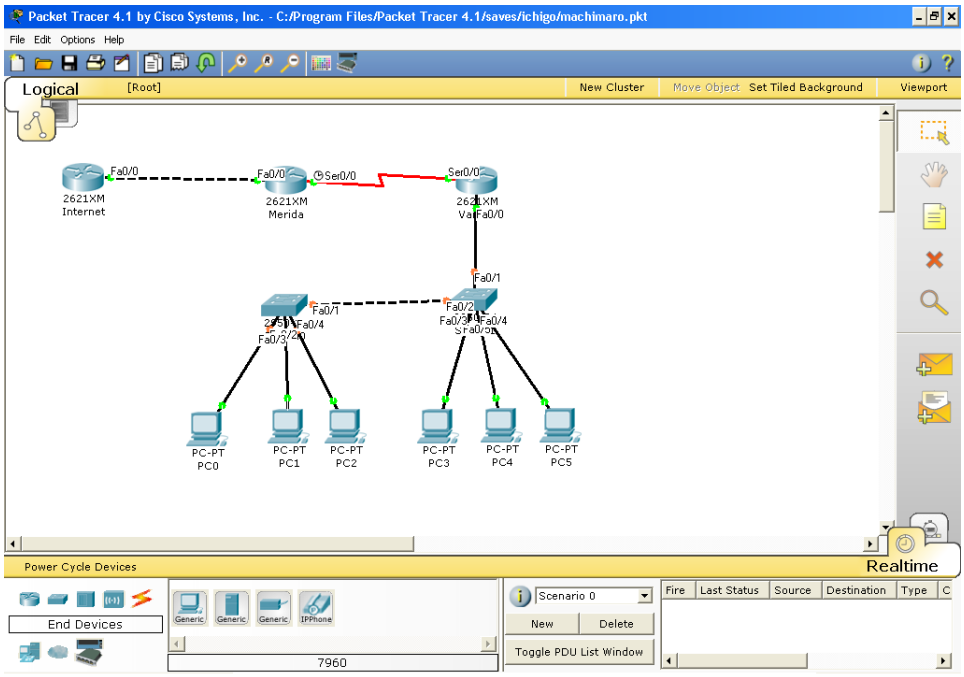
Tento simulátor je produktom spoločnosti Cisco a ponúka výborné a veľmi prepracované zariadenia a prostredie.

Z ponuky zariadení je možné si vybrať zo 4 základných smerovačov (1841, 2620XM, 2621XM, 2811) a z dvoch plne modifikovateľných. Pri konfigurácii každého zmerovača (ale aj hociktorého iného zariadenia) je zobrazený obrázok ako dané zariadenie skutočne vyzerá. Tu sú vidieť miesta na prídavné moduly, kde si používateľ môže vložiť ľubovoľný modul.

Ponuka modulov je skutočne široká. Sú tu k dispozícii desiatky modulov pre rôzne druhy spojenia ako napr. dial-up, koaxiálne, ethernetové, sériové, optické a wireless. Väčšina modulov obsahuje viaceré varianty s rôznymi konektormi.

Ďalej sa tu nachádzajú prepínače typu a 2950-24, 2950T, 2960 a dva modifikovateľné. Samozrejme nechýbajú zariadenia ako rozbočovač a most.

Rozšírením v novších verziách je wireless access point (Linksys WRT300N) a k nemu



Obr. 1.21: Ukážka simulátora Packet Tracer

klientské zariadenia. Pri konfigurácii access point-u je dostupné aj gui prostredie rovnaké ako web rozhranie pri reálnom zariadení.

Z ponuky koncových zariadení je možné si vybrať štandardné pc, ktoré je možné využiť aj ako dial-up klienta, alebo wireless klienta. Tiež je tu implementovaná možnosť na terminálové spojenie a webový prehliadač. Ďalším koncovým zariadením je server, z ktorého je možné poskytovať služby ako :

HTTP - tu si používateľ môže nadefinovať jednoduchú HTML stránku

DHCP - služba na prideľovanie IP adries

TFTP - poskytuje 12 rôznych verzií IOS-u

DNS - služba na prekladanie adries na doménové mená

Ak chce používateľ vytvoriť detailnejšiu topológiu, má k dispozícii aj sieťovú tlačiareň a IP telefón. Pri simulovaní WAN zapojení je možné použiť prepínač na Frame Relay, DSL a koaxiálne káble.

Na prepojenie zariadení sú k dispozícii všetky druhy káblov. Ak si používateľ nie je istý, môže použiť automatický výber, kde mu simulátor vyberie najvhodnejší typ spojenia medzi dvoma zariadeniami. Pri wireless prepojení sa vytvorí viditeľné spojenie medzi klientom a access point-om. Novinkou je pripojenie na konzolu zariadenia.

Skoro všetky z uvedených zariadení majú možnosť výmeny modulov pre pripojenie do sieťovej topológie. Ku každému je dostupný konfiguračný sprievodca, alebo aj špecifické rozhranie k danému zariadeniu ako CLI, Web a iné.

Program má implementovaný aj indikátor funkčnosti zapojenia medzi dvoma zariadeniami. Taktiež je tu implementovaná výborná funkcia na odhaľovanie chýb v zapojení. Je ňou spomalenie času, kedy je možné sledovať ako postupne prechádzajú sieťou jednotlivé pakety. Pri každom je možné si pozrieť vložené údaje na jednotlivých vrstvách tak ako sú definované v OSI modeli.

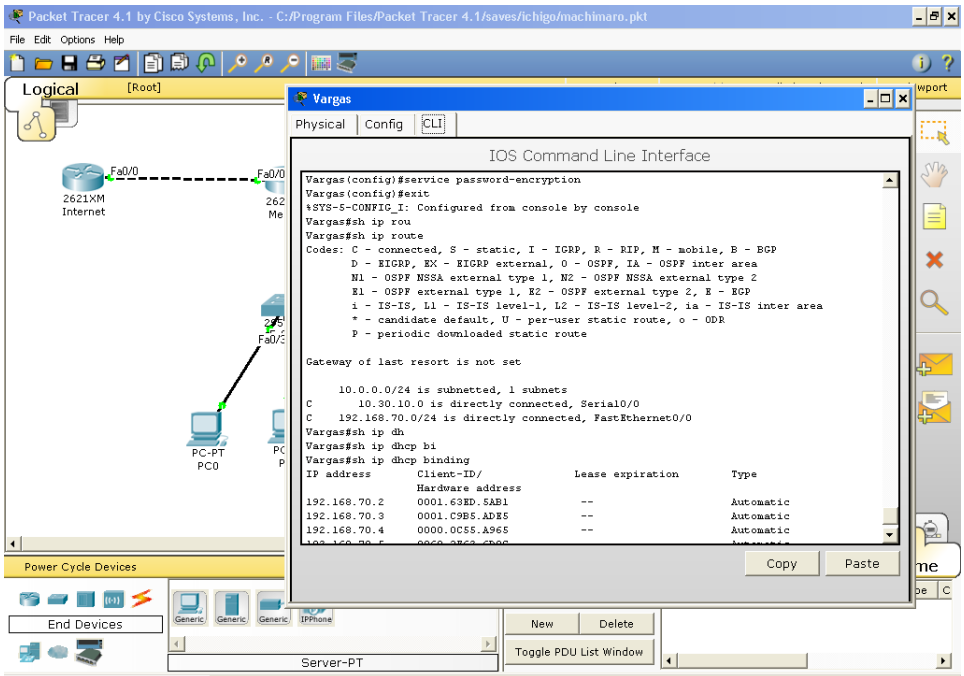
Ďalšou veľmi zaujímavou funkciou je testovanie správnosti zapojení a konfigurácie. K simulátoru je dostupných cez 30 zapojení k jednotlivým semestrom CCNA. Ku každému zapojeniu je popis požiadaviek na konfiguráciu, percentuálne vyhodnotenie splnených požiadaviek a doba trvania. Na záver si používateľ môže pozrieť sumarizáciu, kde presne vidí, ktoré časti splnil a ktoré nie.

Simulátor nie je ani pri zložitejších zapojeniach (okolo 20 rôznych zaradení) náročný na systémové prostriedky.

Implementácia CLI je veľmi dobrá ako aj celkový dojem z prostredia tohoto simulátora. Aplikácia obsahuje aj drobné chyby, ale tie nemajú vplyv na funkcionalitu prostredia. Snaha autorov o čo najväčšie priblíženie sa realite je značná, ale treba si uvedomiť, že sa stále jedná iba o simuláciu, ktorá sa môže odlišovať od reálnej implementácie.

Hlavné okno Packet Tracer (obr. 1.21) obsahuje topológiu siete a ovládacie prvky. Na spodnej časti sa nachádzajú prípadné nové inštancie prvkov ako sú smerovače, prepínače a podobne. Packet Tracer umožňuje dokonca použiť aj bezdrôtové technológie WiFi.

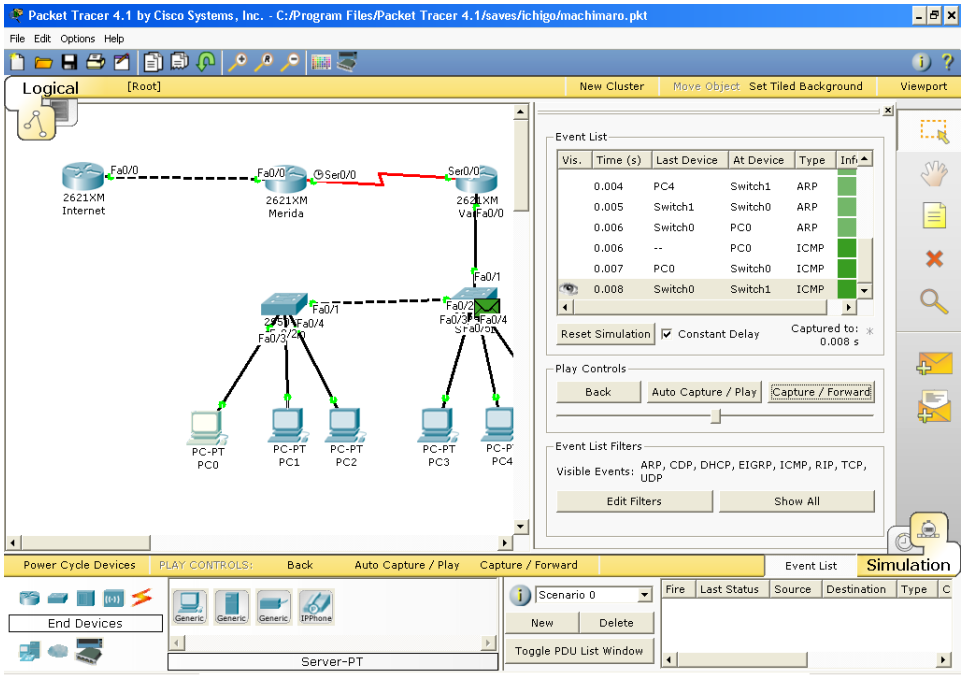
Po vytvorení topológie máme možnosť jednotlivé prvky nastavovať. Konfigurácia je podobná ako na skutočnom zariadení, čiže prostredníctvom CLI (obr. 1.22).



Obr. 1.22: Ukážka konfigurácie zariadenia v simulátore pomocou CLI v aplikácii Packet Tracer

Dobrou vlastnosťou simulátora je možnosť krokovať sieťovú komunikáciu a sledovať tok

dát. Simulácia teda môže prebiehať v reálnom čase, alebo po krokoch.



Obr. 1.23: Ukážka krokovania simulácie v aplikácii Packet Tracer

1.3 Dynamips

Emulácia duplikuje funkcionalitu jedného systému na inom (hostiteľskom) systéme. Výsledný efekt je ten, že sa sa hostiteľský systém tvári ako systém ktorý emuluje. Emulácia je zameraná na vonkajšie vlastnosti emulovaného systému na rozdiel od simulácie, ktorá sa stará práve o vnútorný stav systému. Z počítačového hľadiska emulácia znamená vytvorenie prostredia pre beh emulovaného systému.

Simulácia je spôsob, pri ktorom sa simulátor snaží tváriť ako simulovaná vec tak, aby mal všetky jej vonkajšie vlastnosti.

Dynamips je emulátor Cisco zariadení. Tento emulátor je schopný emulovať procesorové architektúry, ktoré firma Cisco používa vo svojich smerovačoch. Medzi emulované architektúry patrí powerpc a mips64. Dynamips vytvára prostredie podobné fyzickému hardvéru softvérovo a tým umožňuje spúšťať originálny Cisco sieťový operačný systém IOS. V súčasnosti dokáže dynamips emulovať tieto typy Cisco smerovačov: 1700,7200, 3600, 3700 a 2600.

Na emuláciu počítačovej architektúry využíva dynamips JIT kompilátor, ktorý umožňuje za behu prekladať mips64 alebo powerpc inštrukcie do inštrukcií natívnych pre procesorovú architektúru, na ktorej beží emulátor. Táto technika umožňuje oveľa efektívnejšie využitie zdrojov hostujúceho systému. Problémom kompletnej emulácie cieľovej procesorovej architektúry je to, že emulovanou musí byť aj nop inštrukcia a tým pádom je hosťujúci počítač zaťažovaný emulovaním aj keď smerovač nerobí nič.

Medzi v súčasnosti podporované vlastnosti Cisco smerovačov patria

- DRAM a SRAM pamäť
- Non–Volatile pamäť (NVRAM)
- Signetics SCN 2681 DUART (C7200 Console a AUX porty)
- National Semiconductors NS16552 DUART (C3600/C3700/C2600 Console a AUX porty)
- Dallas DS1620 Temperature Sensors a Voltage Sensors, umožňujúce C7200 Environmental Monitor správne pracovať
- NMC93C46 Serial EEPROM
- štartovacia pamäť typu flash 8 Mb (Intel 28F016SA)
- Galileo GT64010/GT64120/GT96100 PCI rozhrania, DEC 21x50 PCI mosty
- PCMCIA ATA emulácia diskového rozhrania

Dynamips podporuje nasledujúce hardvérové karty pre Cisco smerovače

- Cisco 7200 Port Adapters (PA)
  - FastEthernet karty typu "C7200-IO-FE" a "PA-FE-TX" založenú na DEC21140 čipe
  - FastEthernet karty typu "C7200-IO-2FE" a "PA-2FE-TX" založenú na i8254x čipoch
  - GigabitEthernet karty typu "C7200-IO-GE-E" a "PA-GE" založenú na Intel i8254x čipoch
  - Ethernet karty typu "PA-4E" a "PA-8E" založenú na AMD Am79c97x čipoch
  - ATM kartu typu "PA-A1" založenú na Texas Instruments Tneta1570 čipe
  - Serial karty typu "PA-4T+" a "PA-8T"
  - POS (Packet over Sonet) kartu "PA-POS-OC3"
- Cisco 3600 (3620,3640,3660) Network Modules (NM)
  - Ethernet karty typu: "NM-1E", "NM-4E" a "NM-1FE-TX", všetky založené na AMD Am79c97x čipoch
  - Ethernet prepínací modul: "NM-16ESW"
  - Serial card "NM-4T"
- Cisco 2691/3725/3745 Network Modules (NM)
  - FastEthernet karty typu: "NM-1FE-TX"
  - Ethernet prepínací modul: "NM-16ESW"
  - Serial card "NM-4T"

- Cisco 2600 Network Modules (NM)
  - Ethernet karty typu: "NM-1E", "NM-4E" a "NM-1FE-TX"
  - Ethernet prepínací modul: "NM-16ESW"

Výhodou emulácie v porovnaní s Cisco simulátormi ako Boson a Packet tracer je to, že dynamips umožnuje bežať originálny Cisco IOS. Emulované zariadenia na Dynamipse majú presne také isté vlastnosti ako zariadenia skutočné, ešte aj chyby v IOS systéme sú rovnaké. Zatiaľ čo pri simulátoroch, ktoré sa len tvária ako Cisco zariadenia IOS nie je použitý.

Dynamips emulátor v súčasnosti nie je schopný emulovať cisco prepínače a to najmä pre použitie ASIC modulov v nich. Kvôli nedostatku dokumentácie k použitiu týchto modulov nie je bez pomoci firmy Cisco možné implementovať emuláciu aj pre prepínače.

Hardvérové nároky

Počet			Veľkosť	Procesor
smerovačov	študentov CCNA	študentov CCNP	RAM	
20	4	2	3GB	x86 dual core
30	6	3	>4GB	x86_64 dual core
40	8	4	6GB	x86_64 dual core
50	10	5	>7GB	x86_64 SMP viacero procesorov

Tabuľka 1.1: Tabuľka hardvérových nárokov v závislosti na počte a type pracujúcich študentov

Dynagen

Pred spustením programu dynagen je nutné uviesť konfiguráciu (obr. 1.24) pre sieťové prvky. V tomto prípade definujeme niekoľko inštancií smerovačov, ktoré spájame sériovými linkami a pomocou virtuálneho prepínača.

```
[127.0.0.1]

[[3640]]
image = /path/to/image
idlepc = 0x605b98d4
ghostios = True
sparsemem = True

[[router R1]]
slot0 = NM-4T
slot1 = NM-4E
model = 3640
s0/0 = R2 s0/0
f1/0 = NIO_gen_eth:rl0

[[router R2]]
slot0 = NM-4T
slot1 = NM-4E
model = 3640
f1/0 = S1 1

[[router H1]]
slot1 = NM-4E
model = 3640
f1/0 = S1 2

[[router H2]]
slot1 = NM-4E
model = 3640
f1/0 = S1 3

[[ethsw S1]]
1 = access 1
2 = access 1
3 = access 1
~
config [conf][+]                                              #1 13/25,15
-- INSERT --
```

Obr. 1.24: Konfiguračný súbor dynamipsu

```
Reading configuration file...

Network successfully loaded

Dynagen management console for Dynamips
Copyright (c) 2005-2007 Greg Anuzelli

=> list
Name       Type       State      Server          Console
R1         3640       running    127.0.0.1:7200  2000
R2         3640       running    127.0.0.1:7200  2001
H1         3640       running    127.0.0.1:7200  2002
H2         3640       running    127.0.0.1:7200  2003
S1         ETHSW      always on  127.0.0.1:7200  n/a
=>
```

Obr. 1.25: Rozhranie programu dynagen

Po načítaní konfigurácie máme možnosť manažovať zariadenia. Je možné ich zastaviť, odchytávať komunikáciu z ich portov alebo sa ne propojiť pomocou aplikácie Telnet a konfigurovať ich v konzolovom prostredí.

```
*  0 con 0               idle                 00:00:00

  Interface    User               Mode         Idle     Peer Address

Elfen_Lied#sh v
Elfen_Lied#sh ve
Elfen_Lied#sh version
Cisco IOS Software, 3600 Software (C3640-IS-M), Version 12.4(10), RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2006 by Cisco Systems, Inc.
Compiled Wed 16-Aug-06 04:04 by prod_rel_team

ROM: ROMMON Emulation Microcode
ROM: 3600 Software (C3640-IS-M), Version 12.4(10), RELEASE SOFTWARE (fc1)

Elfen_Lied uptime is 3 minutes
System returned to ROM by unknown reload cause - suspect boot_data[BOOT_COUNT] 0x0, BOOT_
COUNT 0, BOOTDATA 19
System image file is "tftp://255.255.255.255/unknown"

Cisco 3640 (R4700) processor (revision 0xFF) with 94208K/4096K bytes of memory.
Processor board ID 00000000
R4700 CPU at 100MHz, Implementation 33, Rev 1.2
1 FastEthernet interface
DRAM configuration is 64 bits wide with parity enabled.
125K bytes of NVRAM.
8192K bytes of processor board System flash (Read/Write)

Configuration register is 0x2142

Elfen_Lied#
```

Obr. 1.26: Konzoly jednotlivých emulovaných smerovačov sú rovnaké ako skutočné

Na obrázku 1.26 máme spustenú inštanciu smerovača. Jeho ovládanie je totožné s konfiguráciou skutočného smerovača.

1.3.1 Graphical Network Simulator 3

(www.gns3.net)

S pomocou nasledujúceho produktu je možné emulovať prepojenie skutočných smerovačov. K tomu sa využíva open source aplikácia Dynamips. Samotné GNS3 je len grafická nadstavba k tejto konzolovej aplikácii.

Na spustenie emulátora je potrebné mať reálny obraz IOS-u, ktorý sa používa v skutočných Cisco smerovačoch. Pri každom smerovači je možné modifikovať moduly so sieťovými konektormi, ktoré sa do smerovača zapoja. Ďalej je tu možné špecifikovať veľkosti pamätí RAM, ROM a NVRAM.

Prostredie sa skladá z dvoch fáz. Prvou je fáza návrhu, kedy si používateľ môže nadefinovať zariadenia a prepojenia medzi nimi. Následne môže prejsť do emulačnej fázy. Tu si môže postupne, alebo naraz naštartovať smerovače. Pre každý smerovač si môže zo schémy otvoriť konzolové spojenie na konfiguráciu.

Dynamips je schopný emulovať iba určité verzie smerovačov, ale GNS3 ponúka simuláciu aj iných zariadení. Veľmi dôležitým je prepínač. GNS3 ponúka aj rôzne verzie prepínačov, okrem klasického ethernetového aj ATM a Frame relay. Tieto prepínače plnia iba funkciu definovaného prepojenia a nie je ich možné ďalej konfigurovať ako reálne zariadenia.

Prostredie je používateľsky prístupné a intuitívne. Poskytuje detailné špecifikovanie jednotlivých parametrov vstupujúcich do samotného emulátora. Spracovanie je na dobrej úrovni a využívanie externej aplikácie na emuláciu je pre používateľa úplne transparentné. Nevýhodou, ktorú ale nemôžeme pripisovať GNS3 je náročnosť emulácie. Pri jednom zariadení emulátor zaťaží procesor permanentne na 50% a pri dvoch až na 100%

(testy boli robené na Pentium4 3GHz). Vzhľadom na to, že je nevyhnutné mať obrazy IOS-u, nie je tento emulátor vhodný na študijné účely. Skôr nachádza využitie u odborníkov pri testovaní zapojení a správania sa smerovačov.

1.4 Zhodnotenie

Analýzou existujúcich riešení sme došli k záveru, že jediný spôsob ako dosiahnuť úplne vernú simuláciu je jednotlivé prvky siete emulovať, čo ukázali skúsenosti s programom dynamips a GNS3. Tieto programy však majú dve obmedzenia: prvým je náročnosť na výpočtový výkon, druhým je nutnosť vlastniť kópiu skutočného operačného systému pre smerovače Cisco - IOS.

Vzhľadom na to, že dostatočný výpočtový výkon ani IOS firmy Cisco nie je dostupný pre všetkých študentov, navrhujeme implementovať simulátor na centrálnom školskom serveri. Keďže server je majetkom školy, môže sa na simuláciu použiť.

Kapitola 2

Špecifikácia

V tejto kapitole bude popísaná špecifikácia projektu. V rámci kapitoly budú popísané ciele projektu a pomocou prípadov použitia sa popíše jeho funkcionalita.

Nevýhodou väčšiny simulátorov je ich schopnosť napodobňovať skutočné sieťové prvky a sú navyše väčšinou platformovo závislé. Všetky produkty ako Packet Tracer, Boson Netsim alebo RouterSim sú určené pre operačný systém Windows. Ich ďalšou nevýhodou je, že ich simulácia nie je úplne verná. Často krát v nich chýba funkcionalita skutočných sieťových prvkov alebo je ich implementácia zlá. Potom sa môže stať, že študent tieto virtuálne zariadenia nakonfiguruje správne, ale v simulátore nefungujú. Pre začiatočníkov to môže byť veľmi mätúce, pretože nevie, či chyba nastala z jeho strany alebo je na vine simulátor. Práve z toho hľadiska je výhodnejšie použiť nástroje, ktoré simulujú funkcionalitu skutočných prvkov reálnejšie. Najvernejšiu simuláciu dokážeme sprostredkovať emulátorom, čiže programom, ktorý vytvára pre operačný systém zaradenia prostredie rovnaké ako na skutočnom technickom prostriedku.

2.1 Ciele

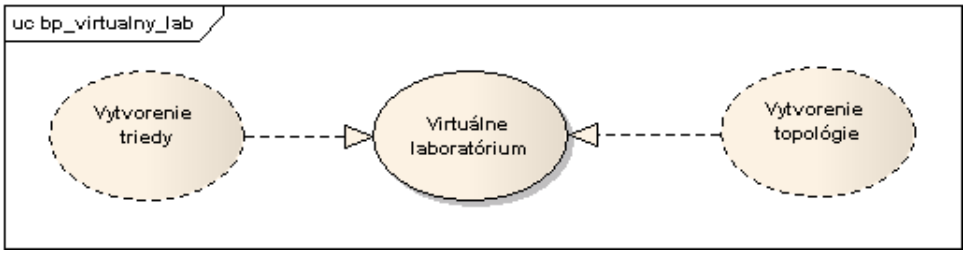
Na základe analýzy sme prišli k záveru, že niektoré emulačné riešenia sú výborné na simuláciu v rámci jedného počítača. Naším zámerom je realizovať riešenie, ktoré by umožňovalo čo najreálnejšiu simuláciu na centrálnom serveri, pričom by umožňoval prístup pre vzdialených používateľov, ktorí by si mohli takto odskúšať reálnu sieťovú topológiu. Predpokladom pre simuláciu topológie je vytvorenie virtuálneho laboratória, ktoré obsahuje centrálny server spolu s možnosťou rezervácie prístupu pre koncových používateľov.

Riešenie v podobe virtuálneho laboratória ma niekoľko výhod. V prvom rade je platformovo nezávislé, čiže používateľ môže použiť ľubovoľnú hardvérovú i softvérovú platformu. Ďalšou výhodou je odbremenenie lokálneho počítača, pretože všetky výpočty sa realizujú na centrálnom počítači. Najväčšou výhodou je vernosť simulácie nakoľko sa skutočné zariadenia emulujú.

Virtuálne laboratórium teda predstavuje centrálny server, na ktorý sa pripájajú používatelia s cieľom simulovať sieťové topológie. Komunikácia so serverom prebieha cez web rozhranie. Používatelia sú rozdelení do viacerých skupín, pričom rozoznávame učiteľov a študentov. I v rámci skupiny študentov existuje diverzita v zmysle potrieb študentov. Podľa zadelenia do skupiny sa určuje podiel výkonu servera.

V projekt sú teda dva ciele (obr.2.1) – vytvorenie virtuálnej triedy a simulácia sieťovej

komunikácie. Virtuálnu triedu bude vytvárať učiteľ a takisto bude vytvárať a definovať sieťové topológie.



Obr. 2.1: Cieľ projektu - virtuálne laboratórium

Predpokladom pre simuláciu komunikácie v sieti bude pre študenta rezervácia lístka, kde si pomocou rozhrania systému vyberie, kedy chce sieťové topológie testovať. Po rezervovaní lístka bude študent oprávnený v danom čase simulovať sieťové topológie zadané učiteľom (obr. 2.2).

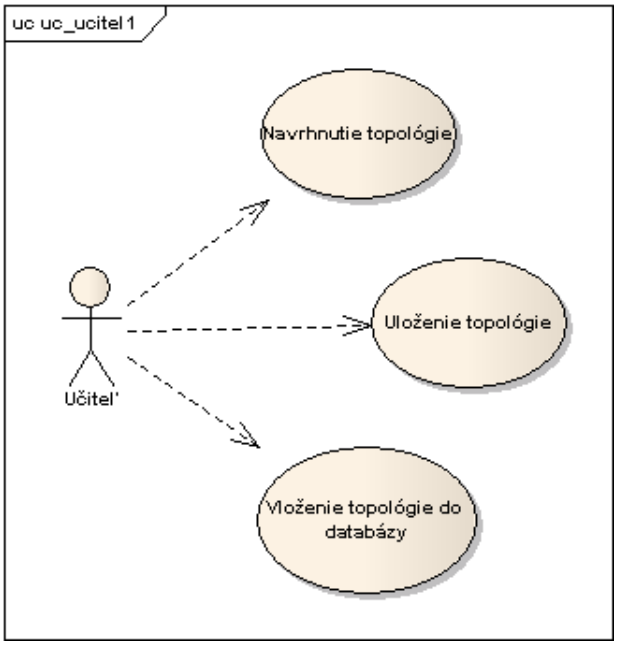
Obr. 2.2: Cieľ projektu - simulácia siete

2.2 Procesy

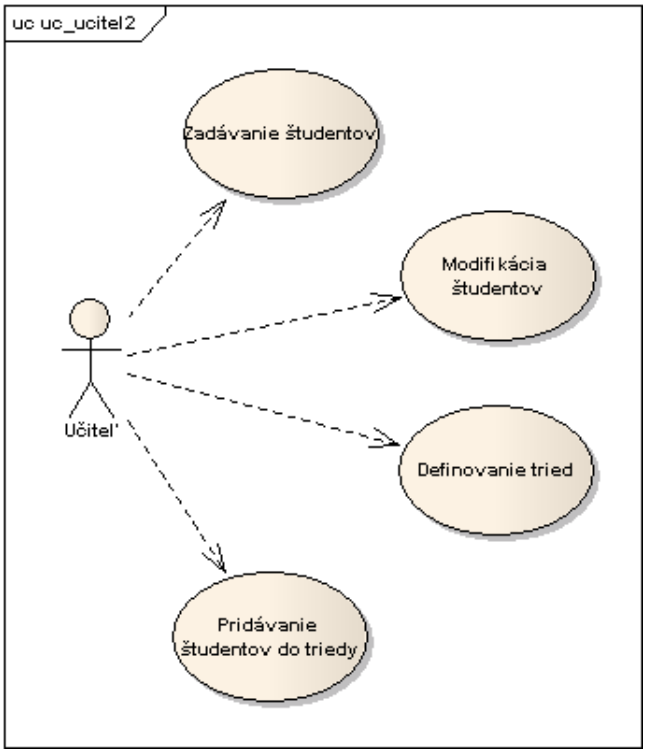
Prvým krokom bude definovanie sieťových topológií, ktoré si budú môcť študenti odskúšať. Učiteľ navrhne a zadá topológiu, uloží ju a napokon vloží do databázy centrálneho servera (obr. 2.3).

Pre vytvorenie virtuálnej triedy je nutné zadávať študentov alebo ich meniť, definovať triedy a pridávať študentov do zadefinovanej triedy (obr. 2.4). Rôzni študenti môžu patriť do rôznych tried, pričom začiatočníci nepotrebujú pre svoje potreby toľko zdrojov servera ako skúsenejší kolegovia. Učiteľ bude podľa potreby zaraďovať študentov do jednotlivých tried, čím im určí, aký výpočtový výkon budú môcť použiť.

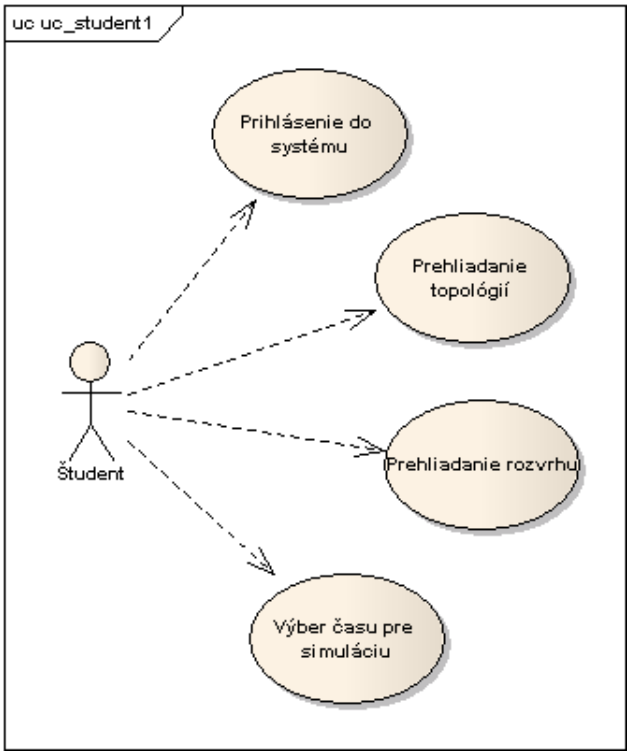
Po vytvorení topológií a po definovaní tried sa môžu oprávnení používatelia prihlásiť do systému. Ich prvým krokom bude prehliadanie aktuálnych topológií a rozvrhu, pomocou ktorého si budú môcť vybrať, kedy si želajú získať prístup na server (obr. 2.5). Server povolí simuláciu viacerých používateľov, ale kvôli hardvérovým obmedzeniam nemôže v jednom momente prebiehať ľubovoľné množstvo simulácií. Systém bude pri každom používateľovi sledovať počet jeho rezervácií, pričom toto číslo nesmie presiahnuť definovaný limit.



Obr. 2.3: Model prípadov použitia - učiteľ - správa topológií

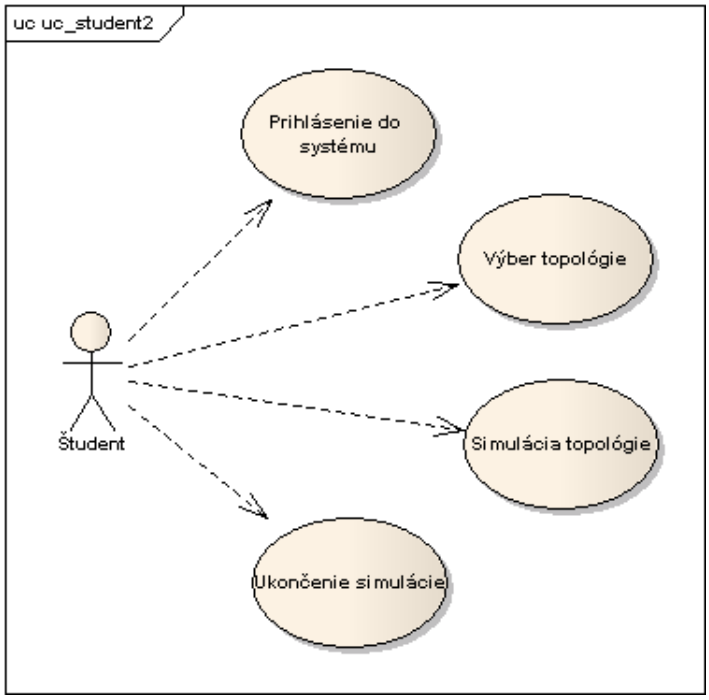


Obr. 2.4: Model prípadov použitia - učiteľ - správa študentov



Obr. 2.5: Model prípadov použitia - študent - rezervácia

V dobe rezervácie sa študent prihlási do systému, vyberie si konkrétnu topológiu, ktorú chce simulovať a inicializuje simuláciu (obr.2.6). Počas simulácie si nebude môcť vybrať inú simuláciu pokiaľ neukončí tú aktuálnu. Po uplynutí prideleného času ukončí simuláciu. Používateľ si počas rezervácie nevyberá konkrétnu topológiu, ale len čas. Takto je študentovi samozrejme poskytnutá väčšia voľnosť.



Obr. 2.6: Model prípadov použitia - študent - simulácia

2.3 Hráči

Študent – študent prichádza do styku so systémom najčastejšie, pretože si rezervuje lístky a vykonáva simulácie, čo zaberá najväčší čas. Študenti sú zaradení do tried, pričom trieda určuje, koľko výpočtového výkonu budú môcť používať.

Učiteľ – rola učiteľa je úzko spojená so správou študentov. Učiteľ má na rozdiel od študentov vyššie privilégia, môžu údaje o študentoch modifikovať a taktiež môže meniť možné sieťové topológie.

Správca - administrátor systému má zo všetkých používateľov samozrejme najvyššie privilégia. Je mu umožnené pracovať s ľubovoľnou entitou v systéme.

2.4 Opis prípadov použitia

UC01 Navrhnutie topológie - zadefinuje sa topológia, pričom fixnými časťami sú počty aktívnych prvkov a ich prepojenie. Voliteľnými časťami ostáva ich konfigurácia a prípadná verzia (napríklad operačný systém smerovača).

UC02 Uloženie topológie - vytvorená topológia sa uloží do formátu použiteľného na centrálnom serveri.

UC03 Vloženie topológie do databázy - podklady pre simulácia sa vložia do databázy centrálneho systému, čím sa sprístupnia pre študentov.

UC04 Zadávanie študentov - pridávanie nových študentov do systému.

UC05 Modifikácia študentov - vyhľadanie parametrov študenta a ich zmena.

UC06 Definovanie tried - vytvorenie tried študentov.

UC07 Pridanie študentov do triedy - zviazanie študentov s virtuálnou triedou.

UC08 Prihlásenie do systému - nakoľko sa jedná o centrálny systém, všetci účastníci musia svoju totožnosť preukázať, čo sa bude diať na základe prístupového hesla. Pomocou tohto hesla sa konkrétnym používateľom sprístupnia vykonateľné operácie.

UC09 Prehliadanie topológií - topológia bude reprezentovaná obrázkom a taktiež bude slovne popísaná.

UC10 Prehliadanie rozvrhu - pomocou rozhrania bude zverejnený rozvrh virtuálneho laboratória respektíve jeho obsadenie.

UC11 Výber času pre simuláciu - študentovi bude ponúknutá možnosť vybrať si termín pre simuláciu k čomu samozrejme musí poznať voľné časy.

UC12 Výber topológie - v čase konkrétnej rezervácie bude mať študent možnosť výberu topológie.

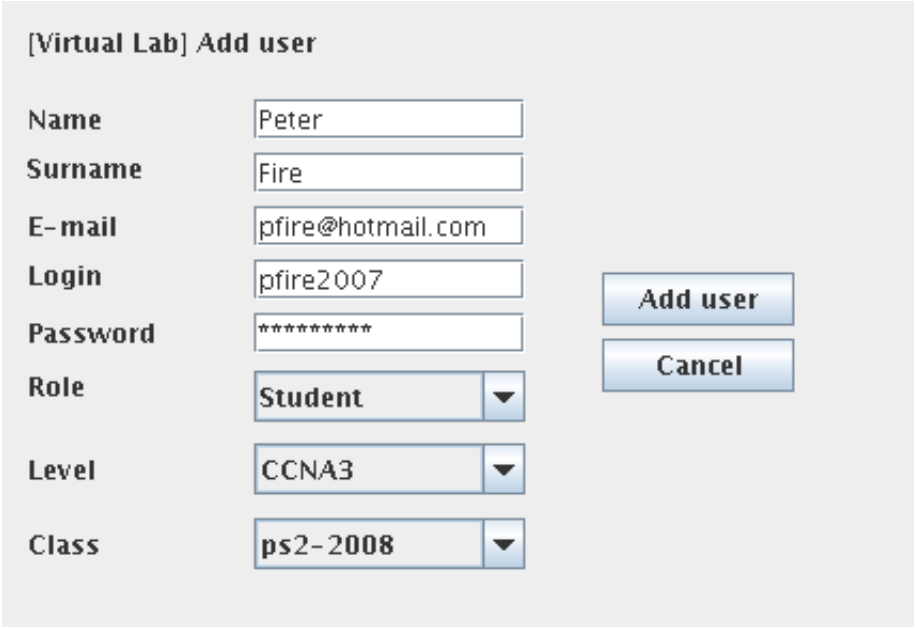
UC13 Simulácie topológie - v čase rezervácie a po výbere topológie ju bude možné simulovať.

UC14 Ukončenie simulácie - po vypršaní času povoleného pre simuláciu sa študent zo systému odhlási a prenechá zdroje iným používateľom. Pred uplynutím časového intervalu bude systémom upozornení na blízke vypršanie prideleného času.

Na ďalších stranách sú podrobne rozpracované vybrané prípady použitia (tab. 2.1 – 2.5) spolu s uvedením obrazoviek používateľského rozhrania (obr. 2.7 – 2.10).

Identifikátor	UC04	
Názov	Zadávanie študentov	
Opis	Učiteľ pridá záznam o študentovi do systému	
Priorita	2 = stredná	
Frekvencia	Mesačne	
Vstupné podmienky	Žiadne	
Výstupné podmienky	Žiadne	
Používatelia	Učiteľ	
Základná postupnosť	**Krok**	**Činnosť**
	1	Učiteľ zadá v systéme požiadavku na zadávanie študentov.
	2	Systém zobrazí formulár pre zadávanie študentov.
	3	Učiteľ vyplní údaje o študentovi do formulára.
	4	Systém skontroluje správnosť zadaných údajov a overí, že sa študent v systéme ešte nenachádza.
	5	Údaje o študentovi sú pridané do systému.
Alternatívna postupnosť	**Krok**	**Činnosť**
	4.a.1	Ak údaje neboli zadané správne, systém upozorní učiteľa, ktorý následne chyby opraví.
	4.a.2	Ak systém zistí, že sa študent v systéme nachádza, upozorní učiteľa a študenta do systému nepridá.
Poznámky		

Tabuľka 2.1: UC04 – Zadávanie študentov



Obr. 2.7: Používateľské rozhranie pre UC04 zadávanie študentov

Identifikátor	UC05	
Názov	Modifikácia študentov	
Opis	Učiteľ upraví údaje o študentovi.	
Priorita	2 = stredná	
Frekvencia	Mesačne	
Vstupné podmienky	Študent musí byť zaregistrovaný v systéme	
Výstupné podmienky	Žiadne	
Používatelia	učiteľ	
Základná postupnosť	**Krok**	**Činnosť**
	1	Učiteľ zadá v systéme požiadavku na zmenu údajov o študentovi.
	2	Systém zobrazí formulár pre úpravu údajov študenta.
	3	Učiteľ zadá identifikačné údaje študenta.
	4	Systém zobrazí aktuálne údaje o študentovi.
	5	Učiteľ upraví údaje o študentovi.
	6	Systém skontroluje správnosť zadaných údajov.
	7	Systém urobí aktualizáciu údajov o študentovi.
Alternatívna postupnosť	**Krok**	**Činnosť**
	6.a	Ak údaje neboli zadané správne, systém upozorní učiteľa, ktorý následne chyby opraví.
Poznámky		

Tabuľka 2.2: UC05 – Modifikácia študentov (Používateľské rozhranie je rovnaké ako pre UC04 – Zadávanie študentov)

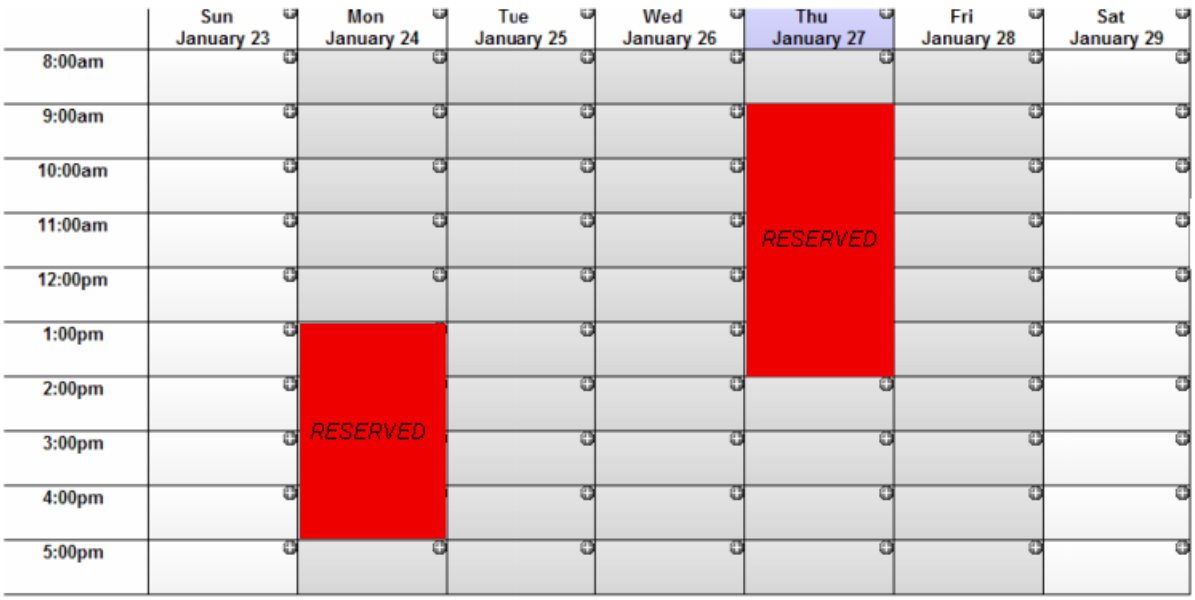
Identifikátor	UC08	
Názov	Prihlásenie do systému	
Opis	Používateľ sa prihlási do systému	
Priorita	1 = vysoká	
Frekvencia	Niekoľkokrát denne	
Vstupná podmienka	Používateľ musí mať vytvorené konto	
Výstupná podmienka	Používateľovi sa zobrazí správa o úspešnosti prihlásenia do systému	
Používatelia	Všetci	
Základná postupnosť	**Krok**	**Činnosť**
	1	Používateľ otvorí web prehliadač a zadá adresu servera.
	2	Na stránke virtuálneho laboratória zadá svoje prihlasovacie údaje.
	3	Systém skontroluje správnosť zadaných údajov.
	4	Používateľovi je povolený prístup do rozhrania systému.
Alternatívna postupnosť	**Krok**	**Činnosť**
	3.a	Ak systém zistí, že používateľ zadal nesprávne prihlasovacie údaje,tak ho upozorní a neumožní prihlásenie do systému.
Poznámky		

Tabuľka 2.3: UC08 – Prihlásenie do systému

Obr. 2.8: Používateľské rozhranie pre UC08 prihlásenie do systému

Identifikátor	UC11	
Názov	Výber času pre simuláciu	
Opis	Študent si vyberie čas simulácie	
Priorita	1 = vysoká	
Frekvencia	Denne až týždenne	
Vstupná podmienka	Študent musí byť zaregistrovaný v systéme	
Výstupná podmienka	Študentovi sa vytvorí časová rezervácia v systéme	
Používatelia	Študent	
Základná postupnosť	**Krok**	**Činnosť**
	1	Študent sa prihlási do systému.
	2	Študent sa pomocou menu systému naviguje na stránku výberu časovej registrácie.
	3	Študent si prezrie aktuálny rozvrh a vyberie si vyhovujúci termín.
	4	Termín registrácie je pridaný do systému.
Alternatívna postupnosť	**Krok**	**Činnosť**
	3.a	Ak v danom čase nie je možné vykonať ďalšiu rezerváciu, ponúkne sa študentovi možnosť vybrať si náhradný termín.
Poznámky		

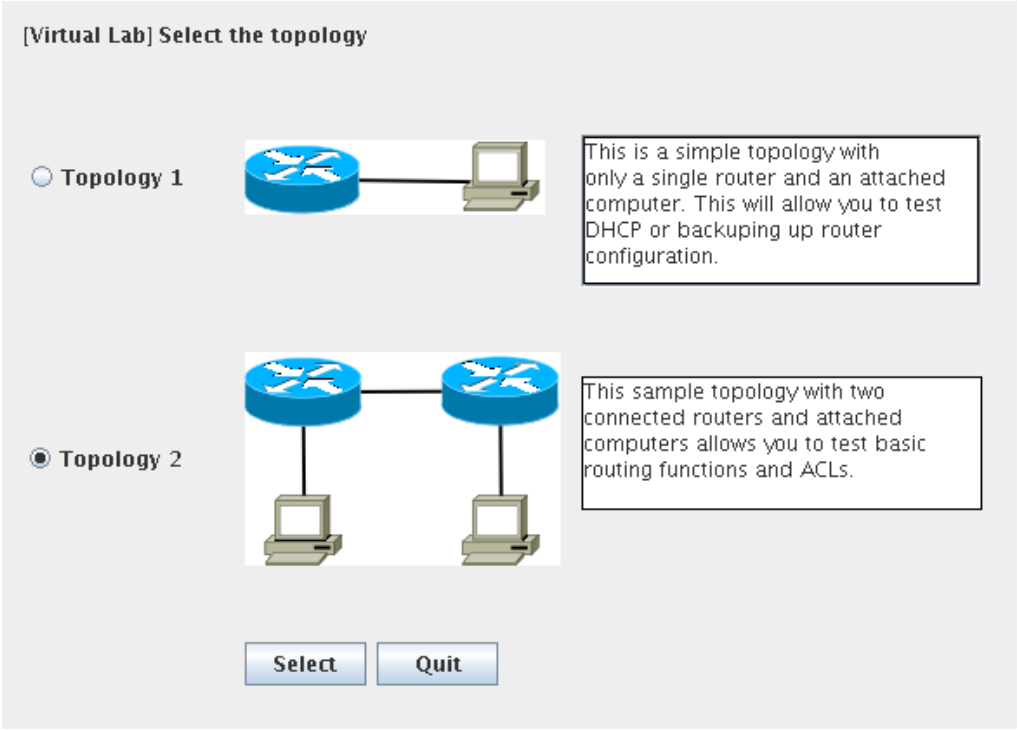
Tabuľka 2.4: UC11 – Výber času pre simuláciu



Obr. 2.9: Používateľské rozhranie pre UC11 výber času simulácie

Identifikátor	UC12	
Názov	Výber topológie	
Opis	Študent si vyberie topológiu pre simuláciu	
Priorita	1 = vysoká	
Frekvencia	Denne	
Vstupná podmienka	Študent musí byť zaregistrovaný v systéme a mať aktívnu rezerváciu	
Výstupná podmienka	Študentovi bude umožnené simulovať zvolenú topológiu	
Používatelia	Študent	
Základná postupnosť	**Krok**	**Činnosť**
	1	Študent sa prihlási do systému.
	2	Študent sa pomocou menu systému naviguje na stránku prehliadania topológií.
	3	Študent si vyberie vyhovujúcu topológiu.
	4	Systém sprístupní študentovi simuláciu zvolenej topológie.
Alternatívna postupnosť	**Krok**	**Činnosť**
	1.a	Neúspešné prihlásenie je používateľovi oznámené a študent je vyzvaný zadať prihlasovacie údaje znova.
Poznámky		

Tabuľka 2.5: UC12 – Výber topológie



Obr. 2.10: Používateľské rozhranie pre UC12 výber topológie

Kapitola 3

Návrh

3.1 Operačný systém

Virtuálne Cisco labolatórium sme sa rozhodli implementovať pod operačným systémom Linux. Na emuláciu rôznych druhov Cisco zariadení použijeme emulátor dynamips a rozhranie dynagen. Vhodné by bolo použiť 64 bitovú verziu Linuxu aby bolo možné používať viac ako 4GB pamäte pre celý systém.

3.2 Funkcionalita

Vlab systém pozná tri druhy používateľov a to učiteľ, študent a administrátor.

Pre študenta by mala byť funkcionalita VLab systému nasledovná:

- umožniť používateľom simulovať zvolenú topológiu
- rezervovať si čas na simuláciu
- zvoliť si doplňujúce informácie k danej topológii verziu IOS a pod.

Pre učiteľa:

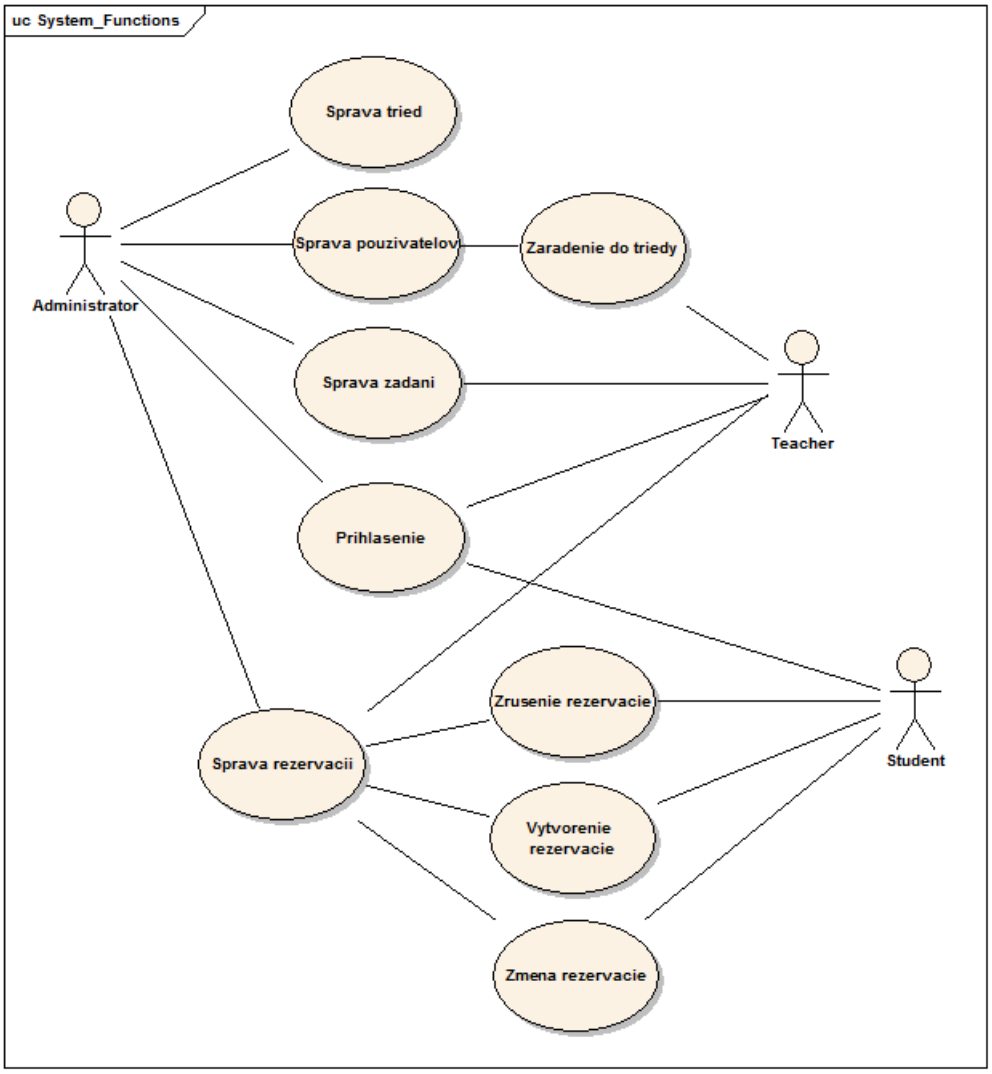
- pridávať jednoduché konfigurácie sieťovej topológie
- pridávať používateľov
- meniť používateľov
- pridávať skupiny používateľov

Pre administrátora:

- meniť základné nastavenie systému

3.3 Rozhranie

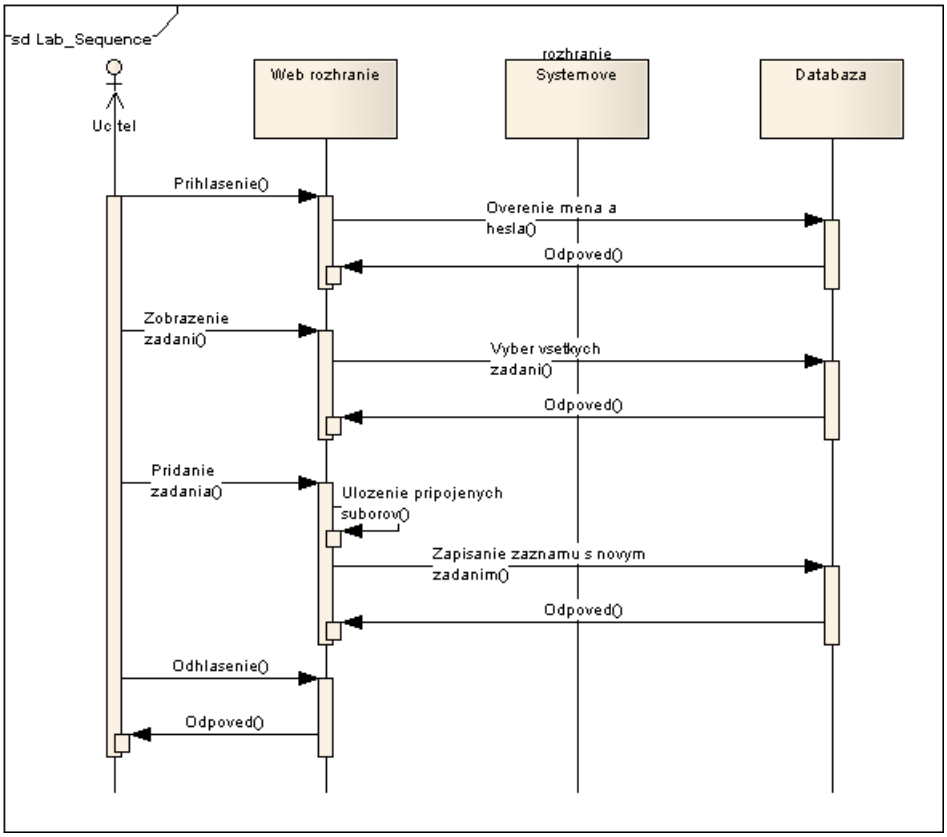
Systém Vlab bude obsahovať web rozhranie, ktoré bude umožnovať správu, prihlásenie, registrovanie času, atď. pre používateľov. Toto rozhranie bude napísané v jazyku php.



Obr. 3.1: Funkcie web rozhrania

Funkcie rozhrania: Používateľ komunikuje so systémom najskôr prostredníctvom webového rozhrania (obr. 3.1). Tu sú mu sprístupnené formuláre podľa jeho prístupových práv. Môže vytvárať rezervácie a počas nich si vyberať zadania na simuláciu. Pre učiteľov je tu možnosť spravovať zadania. Administrátor sa prostredníctvom tohoto rozhrania stará o používateľov, taktiež má plný prístup k ostaným častiam systému, kde môže kedykoľvek zasahovať. Pred prácou sa musí každý používateľ autentifikovať. Podľa toho mu bude pridelená úroveň právomocí. Systém si uchováva väčšinu údajov v databáze, ktoré sú miestami doplnené o externé súbory.

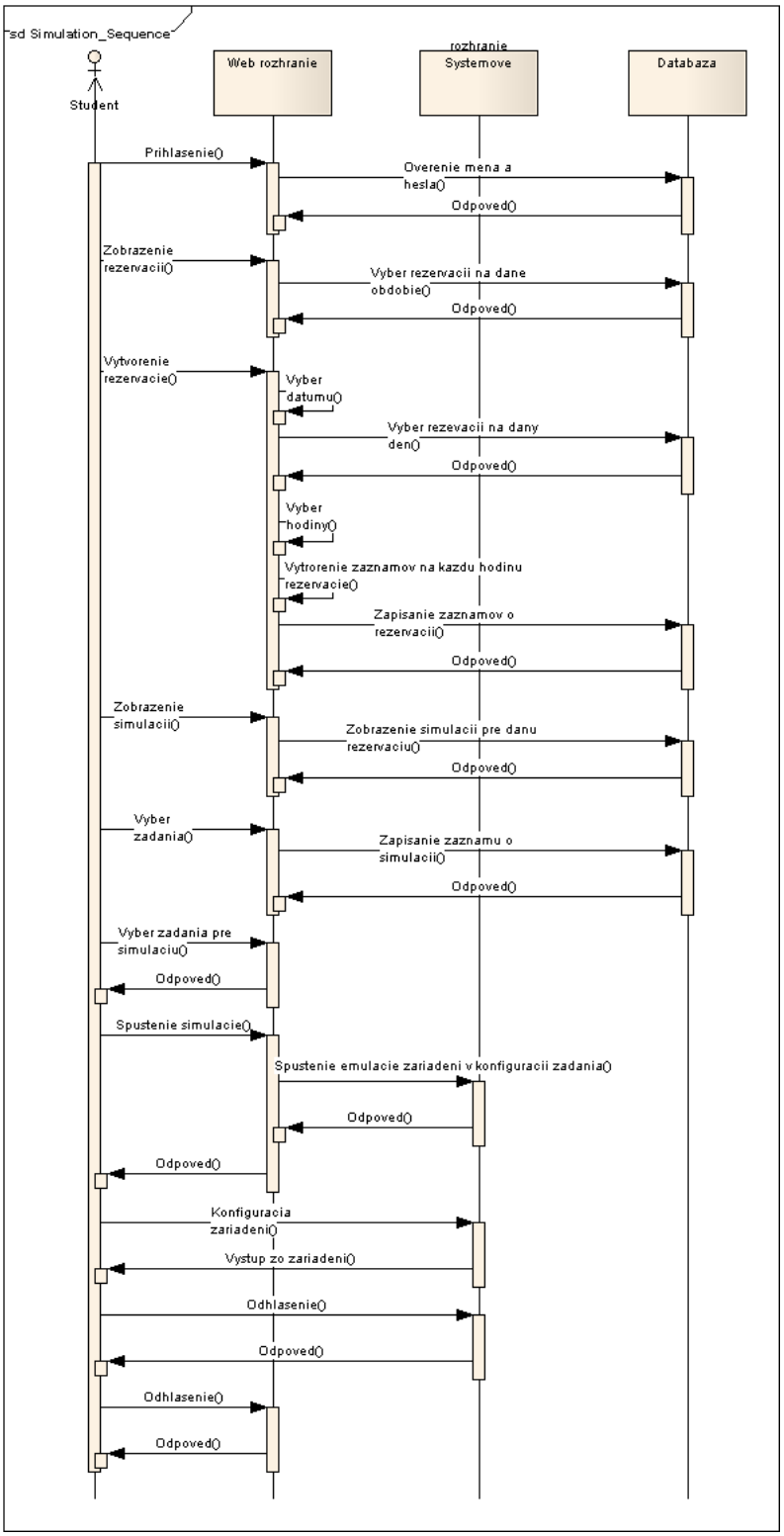
Nasledujúci sekvenčný diagram (obr. 3.2) zobrazuje postup pri pridávaní nového zadania. Pred všetkými aktivitami sa musí používateľ najprv autentifikovať. Potom si zobrazí zoznam zadaní. Ak mu to jeho úroveň prístupu povoľuje zobrazí sa mu možnosť pre pridanie nového zadania. Vyžadovaná úroveň sa prideľuje iba učiteľom a administrátorovi. Po zvolení mu systému zobrazí formulár na potrebné údaje. Popis jednotlivých položiek zadania sa nachádza



Obr. 3.2: Vytváranie virtuálnych topológií učiteľom

v podrobnom popise databázy. Každé zadanie obsahuje aj súbory ako schému zapojenia a dokumenty s poznámkami na vypracovanie. Tieto súbory sú uložené na súborový systém servera. Nakoľko sa jedná o bežné súbory neobsahujúce žiadne privátne informácie, nie je potrebná vyššia úroveň zabezpečenia. Ostatné údaje vložené do formulára sú zapísané do databázy spolu s odkazmi na priložené súbory.

Po úspešnej autentifikácii si študent (obr. 3.3) zobrazí rezervácie. Tu mu systém vypíše všetky časy nakedy si rezervoval prístup na simulovanie. Maximálny počet rezervácii je určený pre každú úroveň študentov zvlášť. Študenti z pokročilejšej úrovne majú možnosť rezervovať si systém viac krát a na dlhší čas. Tieto hodnoty sú nastavené podľa zadaní v danej úrovni a môžu byť kedykoľvek upravené administrátorom. Predpokladajme, že študent zatiaľ nevyčerpal maximálny počet rezervácii. Zvolí si vytvorenie novej rezervácie, kde mu systém ponúkne výber dátumu. Po zvolení systém zobrazí daný deň rozdelený po jednej hodine, kde systém prepočítal a vyznačil hodiny, ktoré sú dostupné na rezerváciu. Študent si vyberie hodinu od ktorej začne jeho rezervácia a systém mu zarezervuje maximálny počet hodín, ktoré sú dostupné a súvislé za sebou. Každá úroveň má totiž definovanú maximálnu dĺžku rezervácie a systém sa pokúsi rezervovať tento počet hodín za sebou v jednom bloku. Pokiaľ nie je možné rezervovať plný počet hodín systém ponúkne rezerváciu iba v dĺžke dostupných hodín. Ak študent nesúhlasí s dĺžkou rezervácie môže sa vrátiť o krok späť a zvoliť si iný deň. Po potvrdení rezervácie systém zapíše rezervované hodiny.

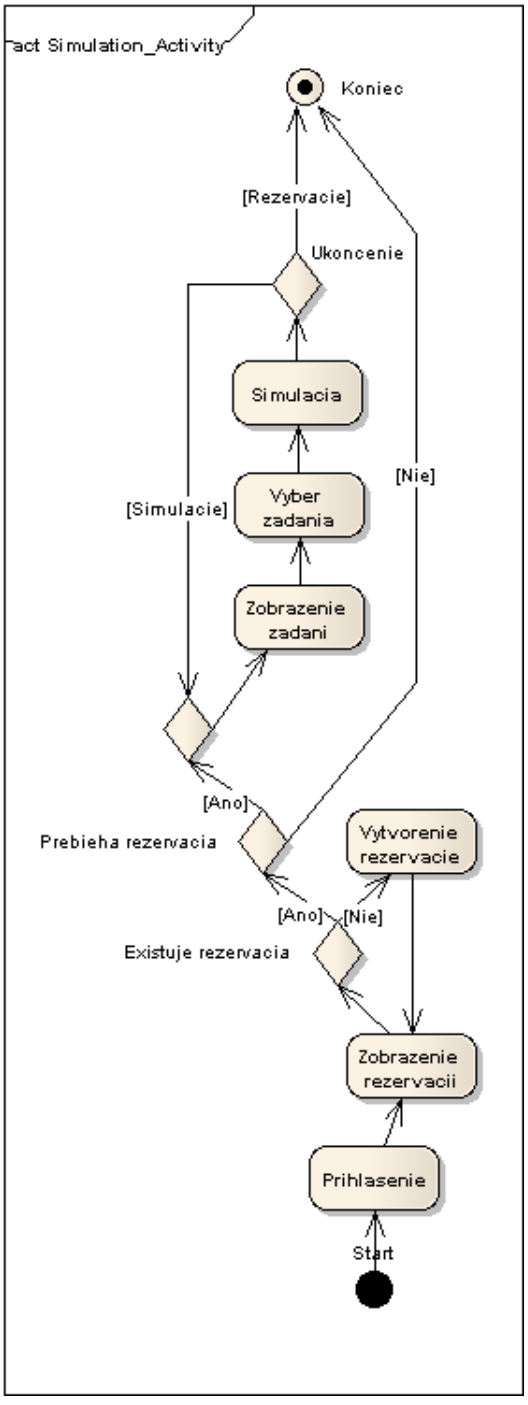


Obr. 3.3: Flow diagram pre študenta

Pokiaľ nenastane čas rezervácie študent môže rezerváciu skrátiť, alebo úplne zrušiť. V prípade, že iný študent zruší rezerváciu čím sa uvoľnia systémove zdroje, ktoré by umožnili rozšíriť jeho rezerváciu, systém túto možnosť študentovi ponúkne, za predpokladu, že nebude prekročný maximálny povolený čas.

V čase rezervácie študentovi pribudne možnosť výberu zadania na simuláciu. Študent si zobrazí zadania a jedno si vyberie. Následne systém vytvorí záznam o simulácii do databázy a zobrazí mu možnosti pripojenia na konfiguráciu jednotlivých zariadení.

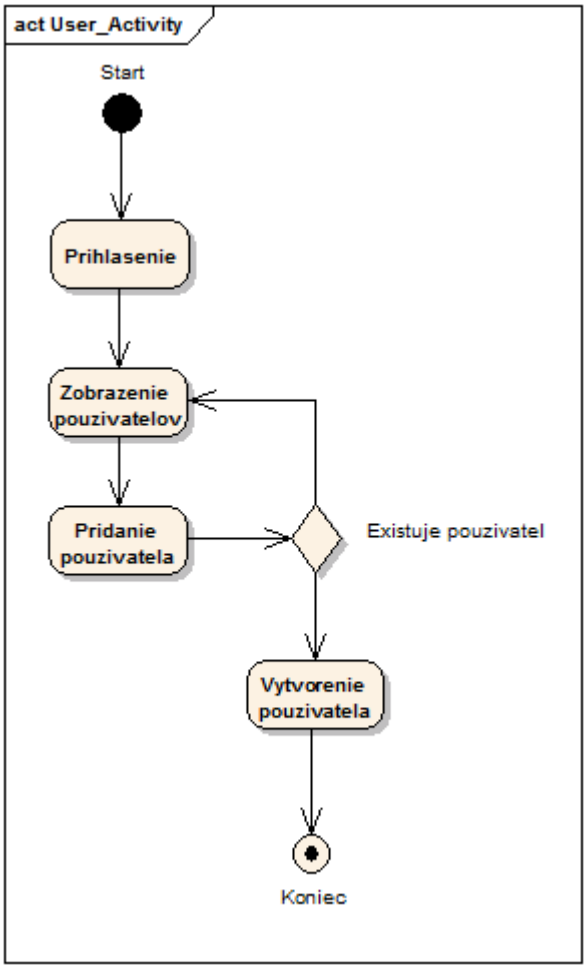
Autorizovaný používateľ s právami administrátora môže spravovať používateľov. Má dostupnú možnosť na zobrazenie všetkých používateľov, odkiaľ môže upravovať aj jednotlivé profily. Tu sa nachádza aj funkcionalita pridania nového používateľa. Po vyplnení formulára systém zapíše do databázy záznam o novom používateľovi. Od tejto chvíle je jeho konto aktívne a je možné sa prostredníctvom neho prihlásiť. Používatelia v skupine učiteľov majú z praktických dôvodov možnosť meniť profily študentov. Napríklad môže nastať prípad, kedy chce učiteľ sprístupniť šikovnému študentovi zadania z vyššej úrovne, tak upraví jeho úroveň definovanú v profile. Alebo študent zabudol heslo, tak mu ho učiteľ umožní zmeniť. Tiež môže nastať situácia, kedy sa študent preradí medzi dvoma triedami, tak učiteľ môže túto zmenu priamo vykonať v systéme.



Obr. 3.4: Postup priebehu simulácie

Nasledujúci diagram znázorňuje postup simulácie študentom (obr. 3.4). Po úspešnej autentifikácii si študent zobrazí rezervácie. Ak ešte žiadnu nemá musí si ju vytvoriť. Ak je čas rezervácie zhodný s aktuálnym časom sú študentovi sprístupnené zadania na simuláciu. Študent si môže vybrať jedno zadanie a začať simuláciu. Ak sa rozhodne zvolenú simuláciu

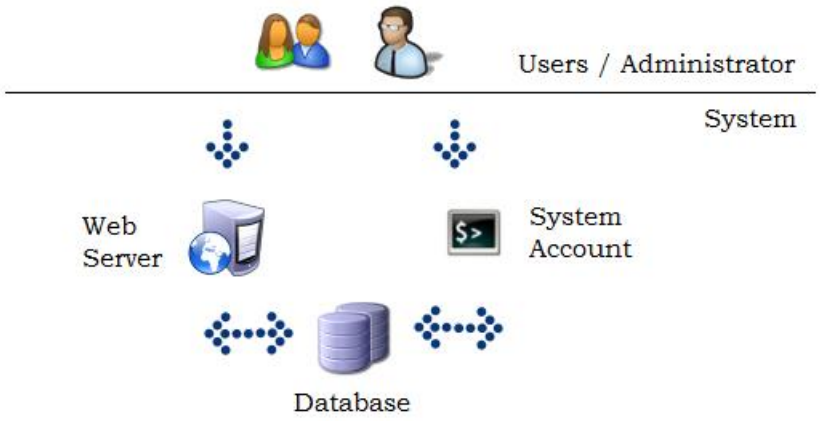
ukončiť môže si vybrať iné zadanie. Systém neobmedzuje študenta na počet simulácii počas jednej rezervácie, ale povoľuje naraz simulovať iba jedno zadanie. Ak uplynie čas rezervácie študent bude odpojený.



Obr. 3.5: Postup pridávania používateľov

V tomto diagrame je v jednoduchosti načrtnutý postup pridávania nového používateľa (obr. 3.5). Na začiatku je vyžadovaná jeho autentifikácia. Potom si zobrazí existujúcich používateľov, kde má možnosť pridať nového, ak mu to úroveň jeho právomocí umožňuje. Po vložení údajov systém overí či daný používateľ už nie je evidovaný v systéme. Ak áno vráti chybovú hlášku. Inak používateľa zapíše do databázy.

Vlab systém (obr.3.6) vytvára virtuálne Cisco laboratórium v ktorom sú študenti, učitelia a administrátori. Učiteľ má na starosti študentov, zadáva im úlohy resp. im prideľuje práva a tým im umožnuje vybrať si jednotlivé topológie na simuláciu. Systém je pre používateľov prístupný iba pomocou web rozhrania cez, ktoré sa používatelia prihlasujú. Používateľ si po prihlásení do systému môže registrovať čas kedy si chce vyskúšať simuláciu. Rezervačný systém funguje tak, že si užívateľ môže registrovať len niekoľko rezervácií dopredu, tak aby nemohol zahltiť systém svojimi rezerváciami.



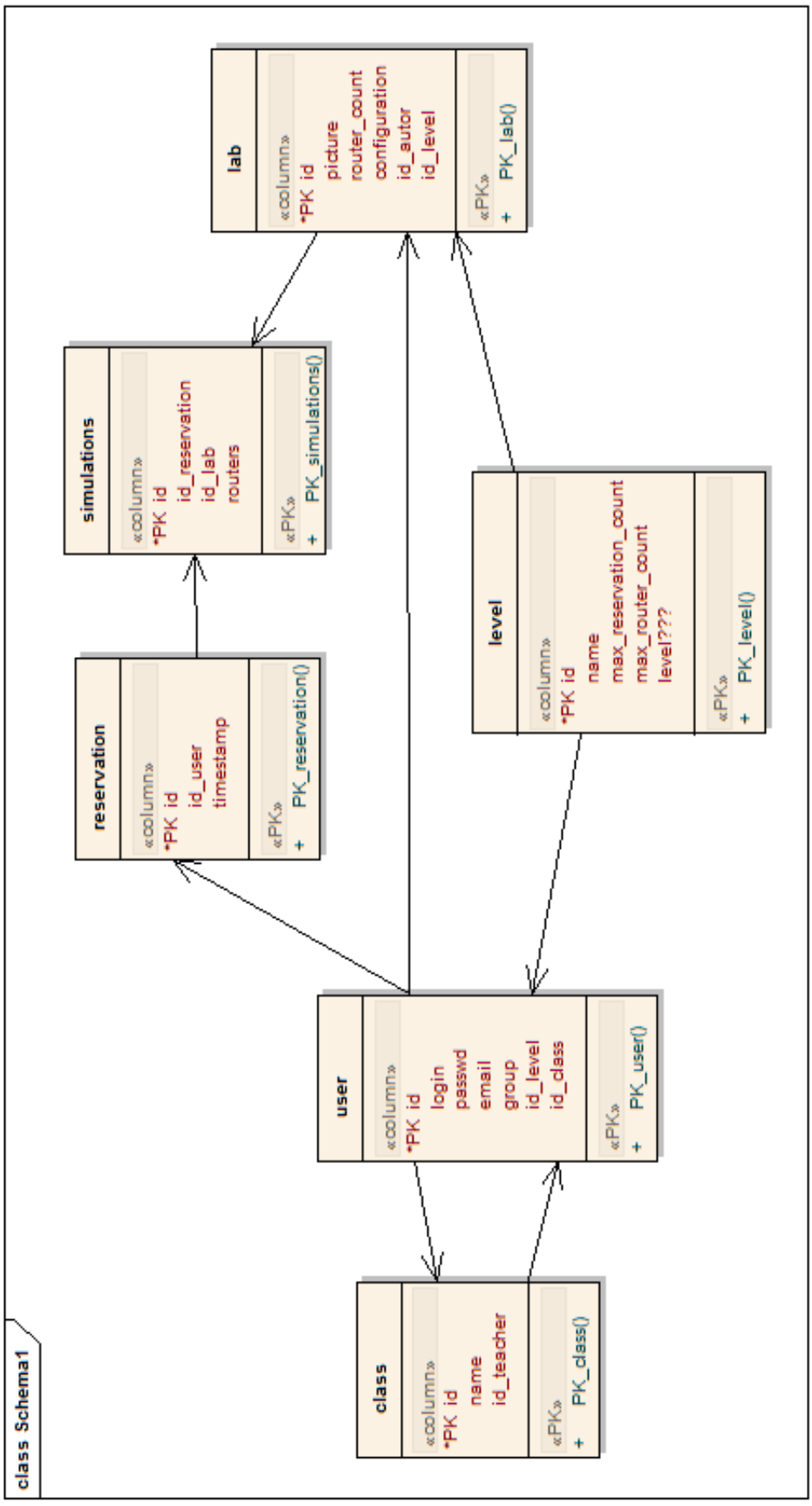
Obr. 3.6: VLab systém

Ak má študent vytvorenú rezerváciu na čas v ktorom sa pripojil, systém mu umožní vybrať si z rôznych virtuálnych topológií, ktoré sú prístupné jeho úrovni oprávnení.

Pri výbere topológie by systém mal minimálne umožniť výber verzie IOS operačného systému pre jednotlivé entity simulovanej topológie. Po naštartovaní simulácie sa používateľ môže pomocou web rozhrania pripojiť na konzoly jednotlivých zariadení. Po vypršaní časového limitu systém užívateľa automaticky upozorní a jeho simuláciu preruší.

3.4 Databáza

Systém ukladá všetky svoje dáta do centrálnej databázy. Tu sú uložené všetky údaje o systéme, užívateľoch a simulovateľných topológiách. Logický model databázy je na obr. 3.7.



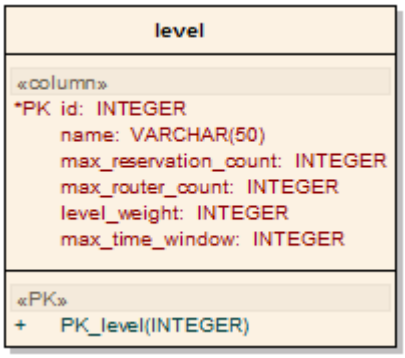
Obr. 3.7: Databáza

Databáza obsahuje tabuľky :

- class
- user
- reservation
- simulation
- level
- lab

3.4.1 Tabuľky

Tabuľka Level



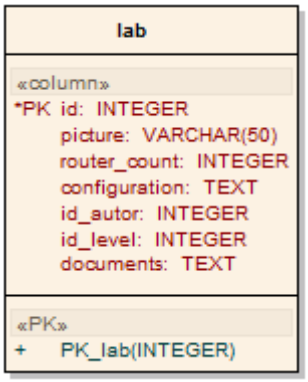
Obr. 3.8: Tabuľka level

Užívatelia sú v systéme rozdelení do viacerých skupín a to podľa toho aké zložité a ako početné simulácie musia vykonávať. Je jasné, že používateľ na úrovni CCNA1 certifikátu má iné požiadavky na VLAB server ako používateľ na úrovni CCNA4 alebo CCNP.

Pre každú úroveň používateľov špecifikujeme tieto údaje:

- **max_router_count** – Významným problémom pri emulácii cisco zariadení je dostatok systémovej pamäťe RAM. Kedže pri emulácii smerovača so 128MB ram musí aj virtuálny smerovač mať k dispozícii 128MB ram. Aby bol systém vlab trvalo funkčný je nutné obmedziť počet zariadení, ktoré sa môžu emulovať súčasne na jednom vlab serveri. Preto je pre každú používateľskú úroveň definovaný maximálny počet zariadení, ktorý vie spustiť na vlab serveri.
- **max_reservation_count** – Kedže náš systém umožnuje registrovať používateľom čas počas ktorého môžu pracovať so svojími topológiami na serveri, je nutné špecifikovať najväčší možný počet rezervácii pre daný level používateľa.
- **level** – je hodnota určujúca prioritu jednotlivých úrovní tak aby topológie patriace nižším úrovniam boli prístupné úrovniam vyšším.

Tabuľka Lab

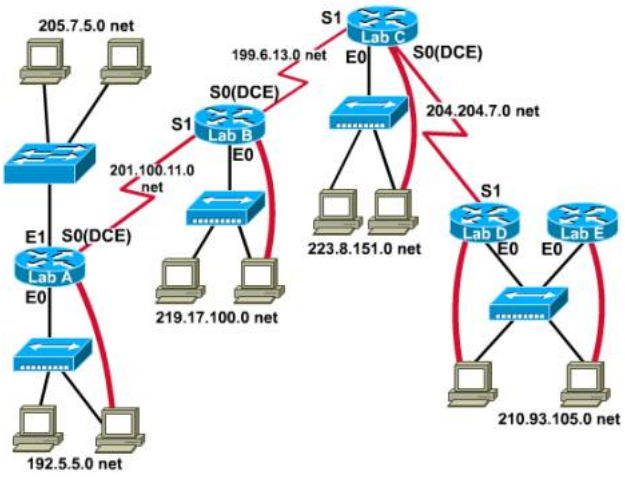


Obr. 3.9: Tabuľka Lab

Táto tabuľka (obr. 3.9) obsahuje všetky možné simulovateľné topológie. Táto tabuľka v sebe obsahuje hodnotu z tabuľky level a to stĺpca s menom level. Toto nám umožnuje priradiť jednotlivé topológie rôznym úrovniam používateľských práv.

Zloženie tabuľky:

- **picture** – Táto hodnota obsahuje konkrétny obrázok zvolenej topológie (napr. obr. 3.10), tak aby si ju bolo možné pozrieť.



Obr. 3.10: Obrázok topológie

- **router_count** – Kedže je na systéme možné spustiť iba určitý počet zariadení, pre každú topológiu si zaznamenávame počet zariadení v nej obsiahnutých. Pri rezervovaní topológie musí systém spočítať množstvo zariadení aktuálne emulovaných na serveri a ak je tento počet plus hodnota pre zvolenú topológiu menšia ako max_router_count, systém spustí simuláciu.

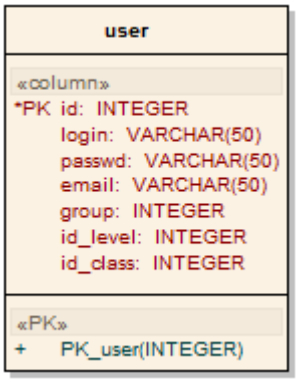
- **configuration** – V tejto položke tabuľky je uložený konfiguračný súbor pre program dynagen. Dlhodobým cieľom projektu je umožniť užívateľské modifikácie tohto konfiguračného súboru, napr. zmenu verzie IOS operačného systému.

- **id_level** – Jedná sa o vzdialený primárny kľúč. Tento kľúč je z tabuľky level a označuje najnižší level pre ktorý je daná topológia prístupná.

- **id_author** – Jedná sa o vzdialený primárny kľúč. Tento kľúč je z tabuľky users a označuje učiteľa, ktorý danú topológiu vytvoril.

- **documents** – Obsahuje dokumenty priložené k danej virtuálnej topológii.

Tabuľka User



Obr. 3.11: Tabuľka používateľov systému

V tejto tabuľke (obr. 3.11) sú uložení všetci používatelia systému.

- **login** – Táto hodnota predstavuje prihlasovacie meno pre používateľa do systému. Kombináciou mena a hesla je používateľ autentifikovaný pri prihlásení do systému.

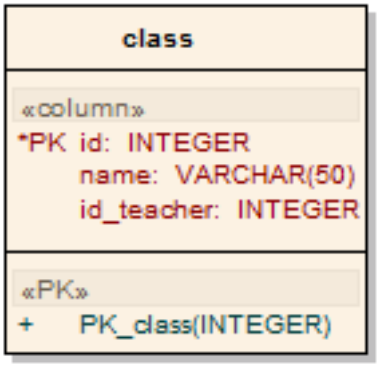
- **password** – Heslo užívateľa v systéme. Heslo je uložené ako odtlačok zadného textu.

- **email** – Používateľov mail.

- **group** – Tento atribút identifikuje používateľskú skupinu. V systéme existujú tri používateľské skupiny a to administrátor, učiteľ a študent. Každá skupina má práva vykonvať rôzne úlohy v systéme.

- **id_level** – Je jednoznáčný identifikátor úrovne používateľských práv. Jedná sa o vzdialený primárny kľúč z tabuľky level.

- **id_class** - Je jednoznáčný identifikátor virtuálnej triedy kam patrí daný používateľ. Jedná sa o vzdialený primárny kľúč z tabuľky class.



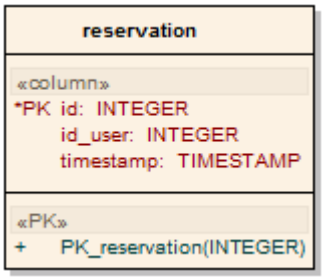
Obr. 3.12: Tabuľka Class

Tabuľka Class

Táto tabuľka (obr. 3.12) obsahuje všetky virtuálne triedy v systéme. Tieto triedy vytvára učiteľ.

- **id** – Toto je primárny kľúč pre tabuľku class.
- **name** – Jedná sa o meno triedy.
- **id_teacher** – Je jednoznáčný identifikátor učiteľa, ktorý vytvoril danú triedu. Jedná sa o vzdialený primárny kľúč z tabuľky users.

Tabuľka Reservations

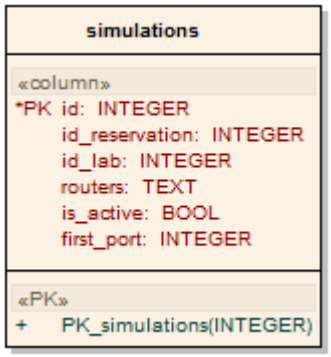


Obr. 3.13: Tabuľka Reservations

Táto tabuľka 3.13 obsahuje užívateľské rezervácie pre daný čas. Pri rezervovaní času sa v tejto tabuľke vytvorí záznam na zvolený čas. Kedže systém je schopný súčastne emulovať iba konkrétny počet cisco zariadení aj počet rezervácií je limitovaný. Maximálny počet rezervácii na čas T je určený úrovňou rezervujúcich používateľov. Kedže každá úroveň používateľov môže mať nastavený iný maximálny počet emulovaných zariadení. Rezervácie sú štandardne generovanné po časových úsekoch rovných 1 hodine, minimálna dĺžka rezervácie je 1 hodina. Pevne zvolený časový úsek sme zvolili kvôli zjednodušeniu práce s kalendárom.

- **id_user** – Je vzdialený primárny kľúč z tabuľky users, ktorý identifikuje používateľa, ktorý vytvoril danú rezerváciu.
- **id** – Je identifikátor jednotlivých rezervácií.
- **timestamp** – Jedná sa o časový údaj ktorý hovorí od kedy je rezervovaná 1 hodina pre používateľa s id_user. V prípade, že používateľ chce vytvoriť rezerváciu dlhšiu ako 1 hodina systém ju rozseká na hodinové úseky.

Tabuľka Simulations



Obr. 3.14: Tabuľka simulations

- **id_reservation** – Je vzdialený primárny kľúč z tabuľky reservations, ktorý identifikuje rezerváciu, ku ktorej daná simulácia patrí.
- **id** – Je identifikátor jednotlivých simulácií.
- **id_lab** – Je vzdialený primárny kľúč z tabuľky labs, ktorý identifikuje virtuálnu topológiu, ku ktorej daná simulácia patrí.
- **routers** – Je dočasne vygenerovaná konfigurácia pre zariadenia v danej viruálnej topológii.
- **is_active** – Kedže jeden používateľ môže mať v jednom momente aktívnu len jednu simuláciu, slúži táto premenná na jej jednoznačnú identifikáciu.
- **first_port** – Toto je číslo portu na ktorom systém spustil virtualne cisco zariadenia. Každé cisco zariadenie je prístupné pomocou telnet programu.

Kapitola 4

Prototyp

Cieľom prototypovania nášho projektu bolo overenie návrhu systému a predvedenie jeho základnej funkcionality. Implementácia prototypu sa skladá z niekoľkých častí:

- inštalácia emulátora dynamips a dynagen,
- vytvorenie databázy,
- vytvorenie proxy servera,
- vytvorenie web rozhrania,
- prepojenie systému.

Architektúra systému sa skladá z vyššie uvedených modulov. Centrálna databáza obsahuje používateľov, ktorí sa môžu pripojiť k systému. Pomocou web rozhrania si registrujú čas, kedy môžu vykonávať simulácie. V čase simulácie sa prihlásia do systému cez web rozhrania a vyberú si topológie, ktoré by si radi odsimulovali. Zároveň spresnia parametre simulácie a pomocou web appletu sa prihlásia na systém. Tento applet im pomocou proxy umožní prístup na jednotlivé virtuálne sieťové prvky. Proxy je použité aj kvôli bezpečnosti, aby sa na server nemohol pripojiť ktokoľvek.

4.1 Dynamips a dynagen

V rámci prípravy programov dynamips a dynagen ich bolo treba nainštalovať na testovací systém. Pri programoch sa použil hypervisor mód, ktorý umožní dynagenu využívať dynamips server. Dynamips je na serveri Vlab vo verzii 0.2.8-RC1 a program dynagen je vo verzii 0.10.1.

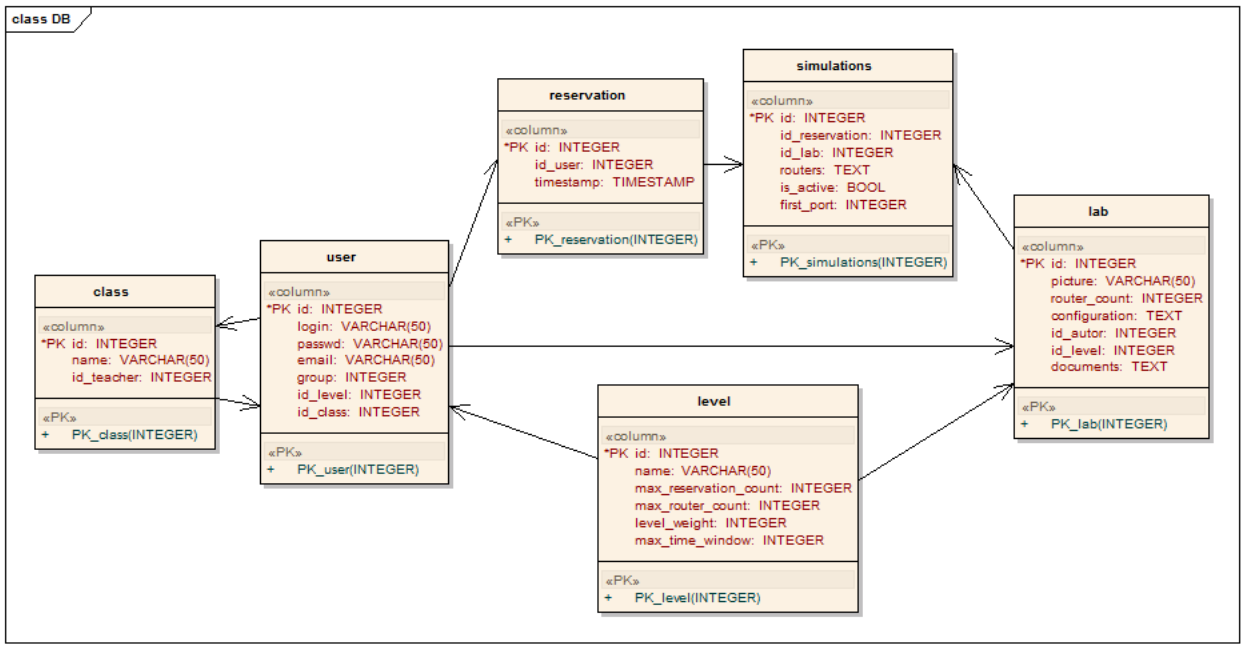
Ďalšou úlohou bolo napísanie niekoľkých vzorových konfigurácií. Tieto konfigurácie reflektujú jednotlivé sieťové topológie. Samozrejme, že jedna konfigurácia môže slúžiť pre odskúšanie viacerých topológií. Pre lepšiu orientáciu boli ku konfiguráciám pripravené aj schémy zapojenia.

4.2 Databáza

Údaje týkajúce sa používateľov a systému sú spravidla udržiavané v databáze. V našom konkrétnom prípade sme použili databázu MySQL verzie 5. Táto databáza je veľmi jednoducho

použiteľná, je šírená s otvoreným zdrojovým kódom, je multiplatformová a navyše má výbornú podporu.

Do databázy boli následne vložené jednotlivé tabuľky tak, ako to vyplýva z hrubého návrhu projektu. Fyzický model databázy je uvedený na obrázku č. 4.1.



Obr. 4.1: Fyzický model údajov

4.3 Proxy server

Proxy server je miestom prepojenia medzi používateľom a reálnym systém. Proxy je použité, pretože sme chceli zamedziť tomu, aby sa mohol používateľ priamo pripájať na virtuálne zariadenia. V našom prípade bude applet kontaktovať proxy, ktoré umožní prístup na jednotlivé zariadenia (smerovač, počítač a podobne).

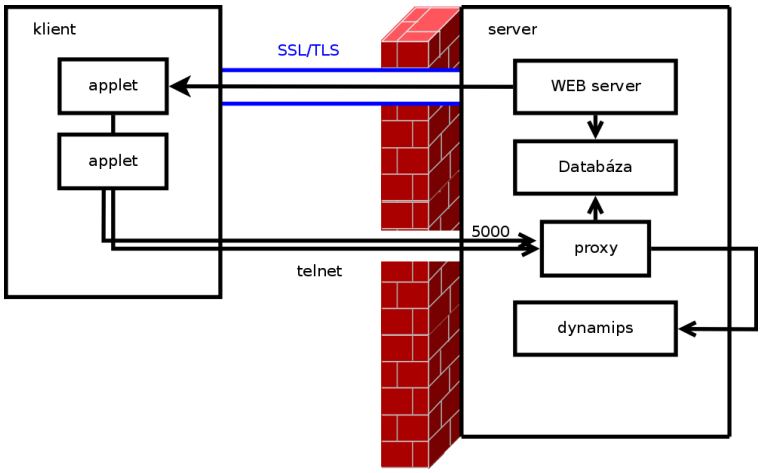
Proxy má niekoľko nastavení:

- Port, na ktorom má byť proxy spustené.
- Cieľový port, na ktorý sa má pripájať.
- IP adresa cieľového servera, v našom prípade to je server Vlab.
- Heslo, ktoré sa bude overovať pri pripojení. Toto heslo však momentálne v prototype nie je využité, ale pri finálnej verzii bude samozrejme využité kvôli overeniu používateľa.

Úlohami proxy servera sú:

- Príjem požiadaviek,
- prepájanie vstupných portov na výstupné,

- zabezpečenie autentifikácie,
- ošetrenie prístupu pri viacerých spojeniach.



Obr. 4.2: Princíp prepojenia prostredníctvom proxy servera

Hlavnou úlohou proxy servera je kontrolovať prístup ku konfiguračným terminálom simulácie. Klient s ním pri pripojení vedie autentifikačný dialóg, ktorý spočíva v odovzdaní jednorázového kľúča vygenerovaného web serverom a jeho overení v databáze. V prípade, že je klient úspešne autentifikovaný, vytvorí sa spojenie s lokálnym portom na ktorom je spustený konfiguračný terminál a proxy prenáša dáta medzí týmito dvoma spojeniami.

Samotné proxy je implementované v jazyku C ako jednoduchý sieťový server, ktorý vytvára nový proces pre každé prichádzajúce spojenie tak, aby bolo možné obsluhovať viacero pripojení od viacerých používateľov zároveň.

Server sa spúšťa s voliteľnými parametrami - port, na ktorom má počúvať a port, na ktorom má kontaktovať SQL server pre overenie autentifikácie. Štandardne počúva na porte 5000/tcp. Ostatné parametre sú implementované priamo v programe, aby sa zabránilo nesprávnej konfigurácii. Program je povinný pripájať sa iba na lokálny počítač.

Prototyp v súčasnom stave nevykonáva kontrolu hesiel pri autentifikácii. Pripojenie je preto povolené akémukoľvek klientovi, ktorý je schopný pripojiť sa a viesť autentifikačný dialóg.

Vzhľadom na kritickosť správnej funkcie proxy servera na bezpečnosť, bude jeho kód preverený vizuálne všetkými členmi tímu a podrobený dôkladnému testovaniu pred jeho uvedením do trvalej prevádzky. Kvôli uvedeným skutočnostiam bude prototyp spustený iba po dobu nevyhnutnú na predvedenie konkurenčnému tímu.

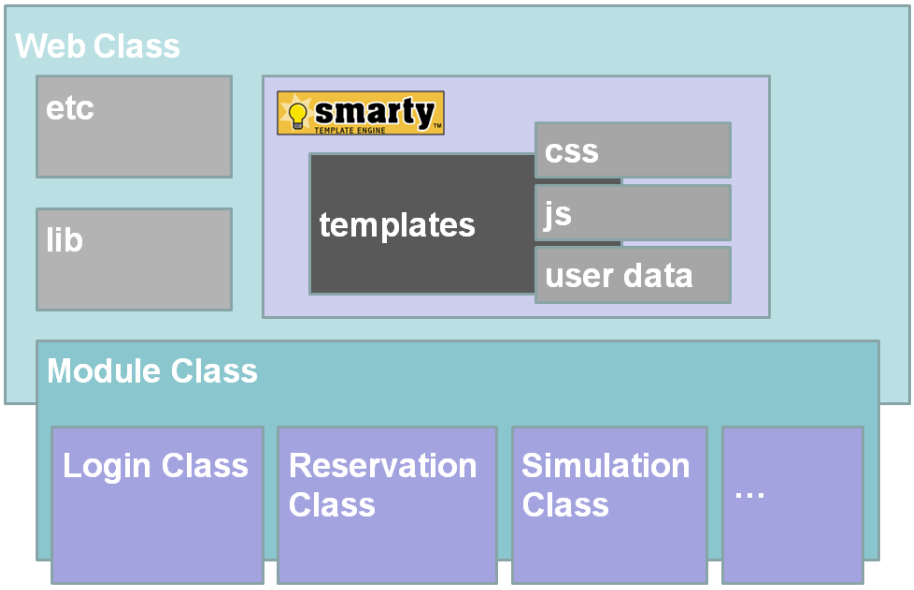
4.4 Web rozhranie

Všetky spomenuté prvky sú pre koncového používateľa v podstate transparentné. Používateľ bude so systémom komunikovať pomocou web rozhrania, ktoré mu umožní:

- Prihlásenie do systému,

- výber času pre simuláciu - registrácia,
- výber topológie pre simuláciu,
- pripájanie na jednotlivé sieťové prvky.

Web rozhranie sme sa rozhodli implementovať pomocou skriptovacieho jazyka PHP spolu s použitím Smarty template. Smarty templates v podstate slúži na separáciu, nakoľko generuje



Obr. 4.3: Smarty templates

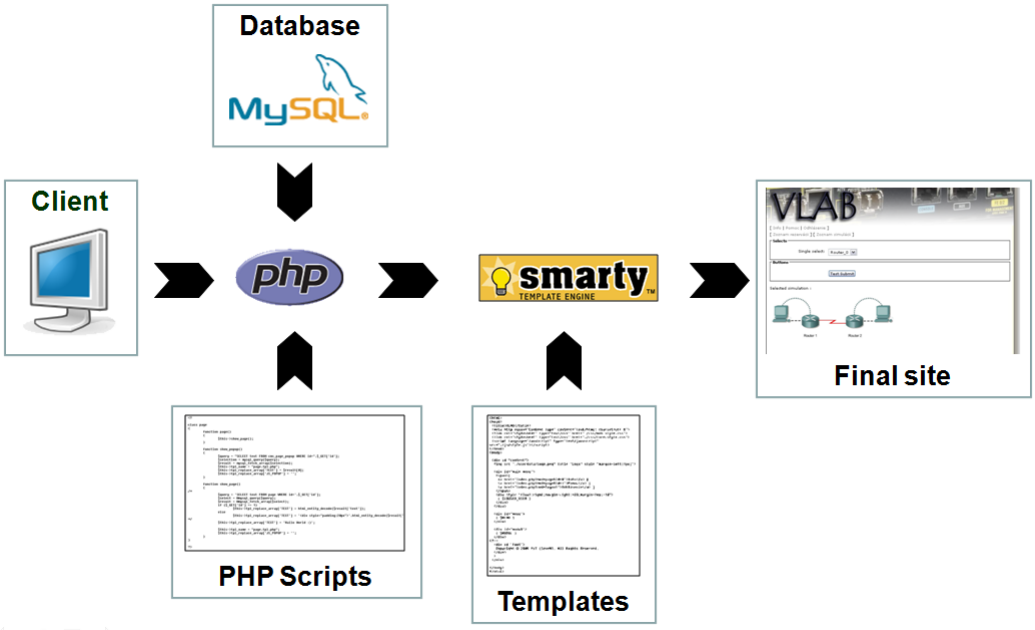
obsah stránky na základe špeciálnych tagov umiestnených v dokumente. Tieto tagy sú potom nahradzované skutočným kódom.

Do generovania výslednej stránky, ktorú uvidí používateľ sú zapojené viaceré moduly systému. V prvom rade sa klient dotazuje na MySQL databázu pomocou PHP skriptov. Na tento výsledok sú potom aplikované Smarty templates a používateľovi je ponúknutá výsledná stránka (obrázok č. 4.4). Súčasný prototyp nám pomocou web rozhrania umožňuje:

- Prihlásenie do systému,
- prezeranie simulácií,
- prezeranie rezervácií,
- štart simulácie,
- prepojenie s appletom.

Dôležitou súčasťou web rozhrania je použitie appletu pre pripájanie používateľov k systému. Tento systému mu v podstate umožní pohodlný a automatizovaný prístup na zariadenia.

Ako implementačný prostriedok sme použili JTA - Telnet/SSH pre JAVA platformu. Tento applet je výhodný z toho hľadiska, že nám poskytuje platformovú nezávislosť.



Obr. 4.4: Web rozhranie

4.5 Implementácia

Každá požiadavka na rozhranie webu sa začne vytvorením inštancie objektu web class. Tato trieda zabezpečuje pripojením potrebných skriptov a vytváranie inštancií modulov.

Základnou funkcionalitou je, že požiada modul class o vytvorenie inštancie modulu login. Tento modul zabezpečuje prihlásenie používateľa. Ak sa daný používateľ už pripojil, overí pravosť jeho údajov a či jeho session ešte neexpirovala. Ak používateľ nie je prihlásený alebo overovanie zlyhalo, vygeneruje formulár s prihlasovacími údajmi.

Ďalej zabezpečí vytvorenie modulu, ktorý si používateľ vyžiadal. Po dokončení modul vráti údaje, ktoré spracoval a nastaví template, ktorý sa má použiť. Tieto údaje sú ďalej poslané systému smarty. Ten zoberie danú šablónu a spracuje ju. Na definované miesta povkladá spracované údaje a výsledný HTML kód vráti ako výsledok.

Keď sú všetky moduly vykonané, ich výsledné HTML kódy sú opäť cez systém smarty nahradené do základnej šablóny webového rozhrania.

Týmto spôsobom vieme veľmi jednoducho oddeliť php kód o designu webu. Tento prístup nám poskytuje väčšiu prehľadnosť kódu a jednoduchú jeho jednoduchú zmenu vo funkčnosti alebo designu.

4.6 Zabezpečenie

Aby sa zabezpečila ochrana prostriedkov reprezentovaných serverom, bolo nutné vykonať viacero bezpečnostných opatrení aj vzhľadom na to, že server je dostupný priamo z Internetu. Preto bol na serveri nastavený paket filter tak, aby umožňoval z vonku pripojenie iba na zabezpečený web (HTTPS), port 5000/tcp na ktorom je spustený proxy server a port 22/tcp pre vzdialený prístup ku konzole ssh určenej iba pre administrátora.

Používateľ má prístup iba k dvom službám. Prvou je bezpečný web, kde má používateľ po prihlásení menom a heslom prístup k svojmu kontu, kde môže vykonávať iba operácie definované v návrhu prípadov použitia. Po spustení simulácie je z web servera stiahnutý a následne spustený applet ktorý je klientom aplikácie telnet. Spolu s aplikáciou sa stiahne aj konfiguračný súbor, v ktorom je uložené jednorázové heslo pre pripojenie k serveru. Používateľ je chránený tým, že nepodpísaný applet nesmie na jeho počítači vykonávať takmer žiadne operácie. Applet sa smie pripojiť iba na server z ktorého bol spustený.

Telnet klient (applet) sa nepripája priamo k serveru, ale najprv k proxy serveru. Klient je pomocou konfiguračného súboru nastavený tak, že sa okamžite autentifikuje jednorazovým heslom. Jednorazové heslo bolo použité preto, lebo telnet komunikácia prechádza cez Internet nešifrovane. Prípadné odchytenie hesla útočníkovi nijak nepomôže, pretože je platné iba pre toto jediné pripojenie. Odchytenie konfiguračných príkazov, ani autentifikačného dialógu sa preto nepovažuje za nebezpečné. Po autentifikácii je spojenie s klientom automaticky presmerované na port vyhradený danému pripojeniu.

Kapitola 5

Implementácia

V predchádzajúcom zimnom semestri 2007/2008 sme v rámci tímového projektu načrtli, akým spôsobom chceme riešiť naše zadanie. Vypracovali sme analýzu na problematiku simulácie počítačových sietí a vytvorili sme hrubý návrh simulačného prostredia spolu s prototypom, ktorý potvrdil, že návrh sa dá uskutočniť. Z požiadaviek a špecifikácie vyšiel záver, že si želáme vytvoriť systém, ktorý bude čo najreálnejšie kopírovať skutočné prostredie a preto sme sa rozhodli postaviť náš systém na báze emulátora dynamips. Zo špecifikácie taktiež taktiež vyplynulo, že celkový systém je nutné doplniť o ďalšie, modulárne časti.

V letnom semestri 2007/2008 máme za cieľ dopracovať návrh nášho riešenia a implementovať také riešenie, ktoré bude spĺňať návrh a rešpektovať špecifikáciu. Táto časť dokumentácie sa člení na niekoľko častí. V úvode popíšeme návrh riešenia a komponenty, z ktorých sa systém ako celok skladá. V ďalšej časti budú taktiež uvedené implementačné prostredia a dôvody, prečo sme sa pre ne rozhodli. V závere zhrnieme celkové výsledky tímového projektu a uvedieme, ktoré časti systému sa nestihli implementovať v plnej miere. Súčasťou dokumentácie bude používateľská i systémová príručka.

5.1 Konfigurácia smerovačov

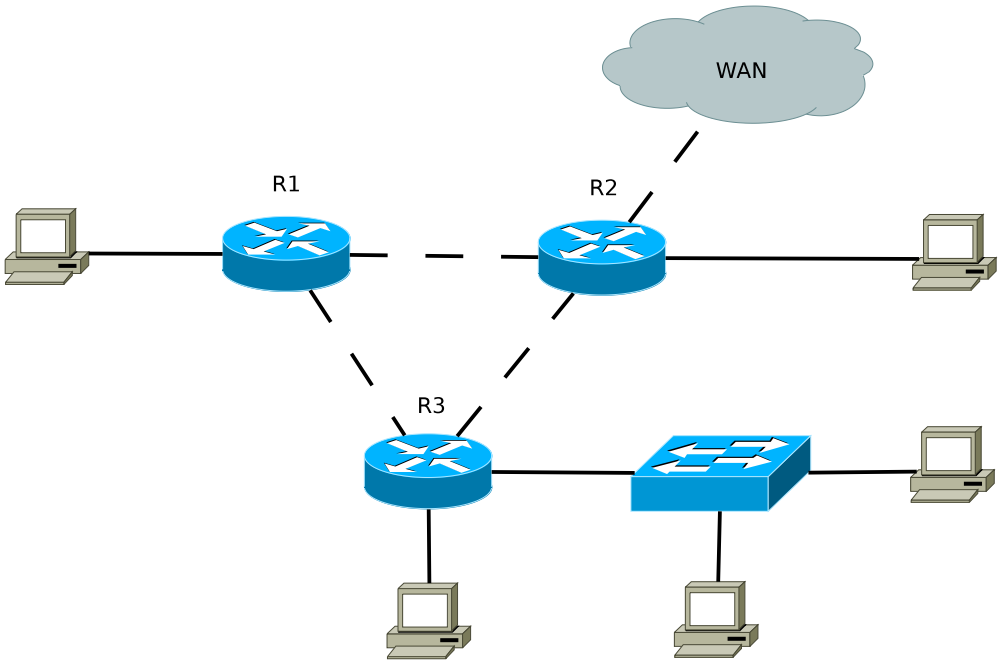
Jednou z najväčších výhod systému dynagen je konfigurovateľnosť. Dynagen v podstate slúži ako front-end pre dynamips. Dynagen nielen vykonáva kontrolu zadaných parametrov, ale rovnako nám umožňuje zapísať konfiguráciu topológie do prehľadného a jednoduchého konfiguračného súboru. Príklad, ako taký konfiguračný súbor môže vyzerať je nasledovný:

```
[[3640]]
image = c3640-is-mz.124-10.image
ram = 128
idlepc = 0x605b98d4

[[router H1]]
model = 3640
slot0 = NM-1FE-TX

[[router R1]]
model = 3640
slot0 = NM-16ESW
```

```
f0/1 = H1 f0/0
f0/4 = H2 f0/0

[[router H2]]
model = 3640
slot0 = NM-1FE-TX
```

Pokiaľ vytvárame tento konfiguračný súbor ručne, tak si pre rôzne simulácie môžeme napísať samostatné konfiguračné súbory. V našom systéme však rovnaké simulácie využívajú rovnakú topológiu, pričom sa líšia v použitých moduloch pre smerovače. Nakoľko chceme dať používateľovi aspoň takú voľnosť, aby mohol pre zvolenú topológiu voliť architektúru smerovača i jeho IOS, tak je nutné požiadavky pre jednotlivé topológie formálne zachytiť a konfiguračné súbory predávané programu dynagen dynamicky generovať.

Ilustrujme si príklad na nasledovnom obrázku 5.1:



Obr. 5.1: Príklad ilustračnej topológie

Prerušovanými čiarami sú načrtnuté spojenia sériovou linkou, plnou čiarou sú vyjadrené Ethernetové spojenia.

Zápis požiadaviek pre jednotlivé smerovače môže vyzerať takto:

```
[[router R1]]
Serial = 2
Ethernet = 1

[[router R2]]
Serial = 3
Ethernet = 1
```

```
[[router R3]]
Serial = 2
Ethernet = 2
```

Ak by sme pomocou takéhoto zápisu určili požiadavky pre jednotlivé smerovače, potom sa pri výbere konkrétnej simulácie obmedzíme len na výber takých modelov, ktoré dokážu splniť tieto požiadavky. Konkrétne v tejto topológii by sme pre smerovač R3 nemohli vybrať radu Cisco 1760, pretože disponuje dvoma slotmi. Jeden slot sa z toho použije pre dve sériové linky, ale ďalší slot môže byť použitý len pre jeden Ethernet port, čo v danom prípade nestačí. Rovnaké komplikácie sú aj s radom Cisco 3620, ktorý opäť disponuje dvoma slotmi, ale musíme vybrať 4-portový Ethernet. Pre daný smerovač nemôže zvoliť FastEthernet, pretože je len jednoportový.

Ak teda máme na serveri uložených niekoľko IOSov, ktoré prislúchajú rôznym architektúram smerovačov, najjednoduchšie bude zapísať ich schopnosti pomocou jazyka XML. Pri výbere simulácie sa potom na základe schopností smerovačov a požiadaviek topológie rozhodneme, ktoré modely ponúkneme používateľovi na výber.

Zápis schopností jednotlivých architektúr smerovačov v jazyku XML je nasledovný:

```
<device>
 <type>Router</type>
 <platform>3600</platform>
 <series>3640</series>

 <ios_list>
  <ios>
   <iosname>c3640-is-mz.124-10.bin</iosname>
   <idlepc>0x605b98d4</idlepc>
  </ios>
 </ios_list>

 <hw_capabilities>
 <slot_family>
  <general type="modular">
   <slots>4</slots>
   <card>
    <name>NM-4E</name>
    <type>Ethernet</type>
    <ports>4</ports>
   </card>
  </general>
 </slot_family>
 </hw_capabilities>
</device>
```

Pre každú sériu dostupných smerovačov sa vytvorí takýto „device" záznam, v ktorom je presne špecifikované akej platformy je a aj séria. Pre každú sériu môžeme mať k dispozícii viacero verzií IOSov, pričom pre každý nie je nutné vytvárať vlastný „device" záznam, ale v

Tabuľka 5.1: Tabuľka chybových kódov skriptu

Kód chyby	Popis chyby
1	Neexistuje užívateľ pre spustenie simulácie
2	Nedá sa naštartovať dynamips alebo dynagen
3	Port je obsadený
4	Neexistuje konfigurácia
5	Nebol zadaný čas ukončenia
6	Nebol zadaný používateľ na ukončenie
7	Nepovolená operácia
8	Nesprávne parametre

rámci séria sa špecifikuje „ios_list", ktorý bude obsahovať názov IOSu a jeho idlepc hodnotu. Hardvérové možnosti série budú špecifikované v položkách „hw_capabilities". Pre sériu sa určí počet slotov cez parameter „slots" a potom sa opíšu karty. Každá karta v type „card" má svoj názov, typ (Ethernet, serial) a príslušný počet portov.

5.2 Štartovacie skripty

O samotné spustenie simulácie sa stará sada scriptov.

5.2.1 Popis skriptov

Startup.sh

Vstupné parametre

1. – začiatočný port simulácie

2. – cesta ku konfiguračnému súboru

3. – dĺžka trvania simulácie

Prvým spusteným skriptom je startup.sh. Tento skript má za úlohu zistiť užívateľa pod ktorým bude spustený skript startup_user.sh . Toto zistuje tak, že sa pokúsi zo zoznamu užívateľov v systéme (zo súboru /etc/passwd) podľa premennej template (táto obsahuje identifikáciu všetkých vyhradených používateľov - spoločnú časť ich mena) sa pokúsi vybrať takého pod ktorým nie sú spustené žiadne programy (a teda nemôže podním byť spustená simulácia). Pokiaľ takéhoto nájde spustí pod ním pomocou príkazu sudo skript startup_user.sh s potrebnými parametrami. Okrem toho spustí skript shutdown.sh na pozadí pod užívateľom root.

Ak niektorý zo skriptov zlyhá, ukončí vykonávanie a vráti návratovú hodnotu v podobe kódu chyby. Návratové hodnoty skriptov sú uvedené v tabuľke 5.1

Ďalej tento skript dá informácie o začatí simulácie do databázy.

startup_user.sh

1. – začiatočný port simulácie
2. – cesta ku konfiguračnému súboru

Tento skript vykoná nasledujúce úlohy

- prejde konfiguračný súbor
- nahradí hodnoty nasledujúcich premenných v súbore za hodnoty, ktoré su aktuálne pre túto inšanciu simulácie

  console – port na ktorom počúva daný smerovač

  host – premenná označujúca že na daný interface je pripojený linuxový uml host. Slovo host nahradí za cestu k unixovému socketu pomocou ktorého sa spoja s emulátorom dynamips

  workingdir – adresár kde sa budú ukladať dočasné dáta potrebné na simuláciu
- spustia inštancie uml linuxu s potrebnými parametrami
- spustia softvérové vde prepínače
- zapíše sa upravený konfiguračný súbor
- spustí simulátor dynamips a počká sa najeho úspešný štart
- spustí generátor dynagen, ktorému pošle upravený konfiguračný súbor

Ako je vidieť z tohoto výčtu, na začiatku skriptu máme náš špeciálne upravený konfiguračný súbor, ktorý upravujeme až na konci dáme dznagenu s ním kopatibilný súbor s parametrami pre danú inštanciu simulácie.

shutdown.sh

1. – užívateľ ktorému sa ukončí simulácia
2. – za ako dlho sa ukončí simulácia (v sekundách)

Tento skript ukončí simuláciu za daný počet sekúnd. Pomocou neho vieme určiť dĺžku každej simulácie. Kontroluje či je daný užívateľ zo skupiny, ktorá môže spustiť simuláciu.

5.2.2 Inštalácia

Na to aby skripty fungovali je potrebné vytvoriť

- užívateľov simu00-xx ,
- skupinu simu v ktorej všetci budú,
- adresár /tmp/simu (alebo iný avšak treba upraviť premenné v skriptoch).
- mať funkčný UML kernel s userspace a zadať cestu k nemu v skriptoch

- nainštalované uml-tools
- mať nainštalovaný vde_switch
- mať v /etc/sudoers nasledujúce riadky (pokiaľ sa startup.sh spúšta pod užívateľom apache a skripty sú uložené v adresári /opt/scripts/)
  - apache ALL=(%simu) NOPASSWD: ALL
  - apache ALL=(ALL) NOPASSWD: /opt/scripts/shutdown.sh

5.3 Databáza

5.3.1 Fyzický model údajov

Z procesu návrh a implementácie vyplynuli aj niektoré zmeny v databáze. Zmeny sa týkajú nasledovných tabuliek.

Tabuľka User

Do tabuľky pribudli záznamy first_name, last_name, ip, session, last.

first_name - Krsné meno používateľa.

last_name - Priezvisko používateľa.

IP - IP adresa, z ktorej sa prihlásil používateľ.

Session - Značí aktuálnu reláciu používateľa.

Last - Tento parameter určuje predchádzajúcu reláciu používateľa so systémom.

Tabuľka Reservation

Do tejto tabuľky pribudol len jediný záznam, a to unique_str.

unique_str - Toto je unikátny reťazec, ktorý slúži pre autentifikáciu používateľa.

Tabuľka Simulations

V tabuľke pribudli nasledovné záznamy - unique_str, started, uid, auth_hash

unique_str - Rovnako ako v tabuľke reservation, tento reťazec slúži pre autentifikáciu používateľa.

started - Tento záznam určuje, či už bola simulácia spustená.

uid - UID (User Identification) je identifikácia používateľa Unix systému, pod právami ktorého pobeží simulácia.

auth_hash - Autentifikačná hash hodnota, posiela sa medzi web rozhraním a proxy serverom za účelom autorizácie používateľa.

Tabuľka Lab

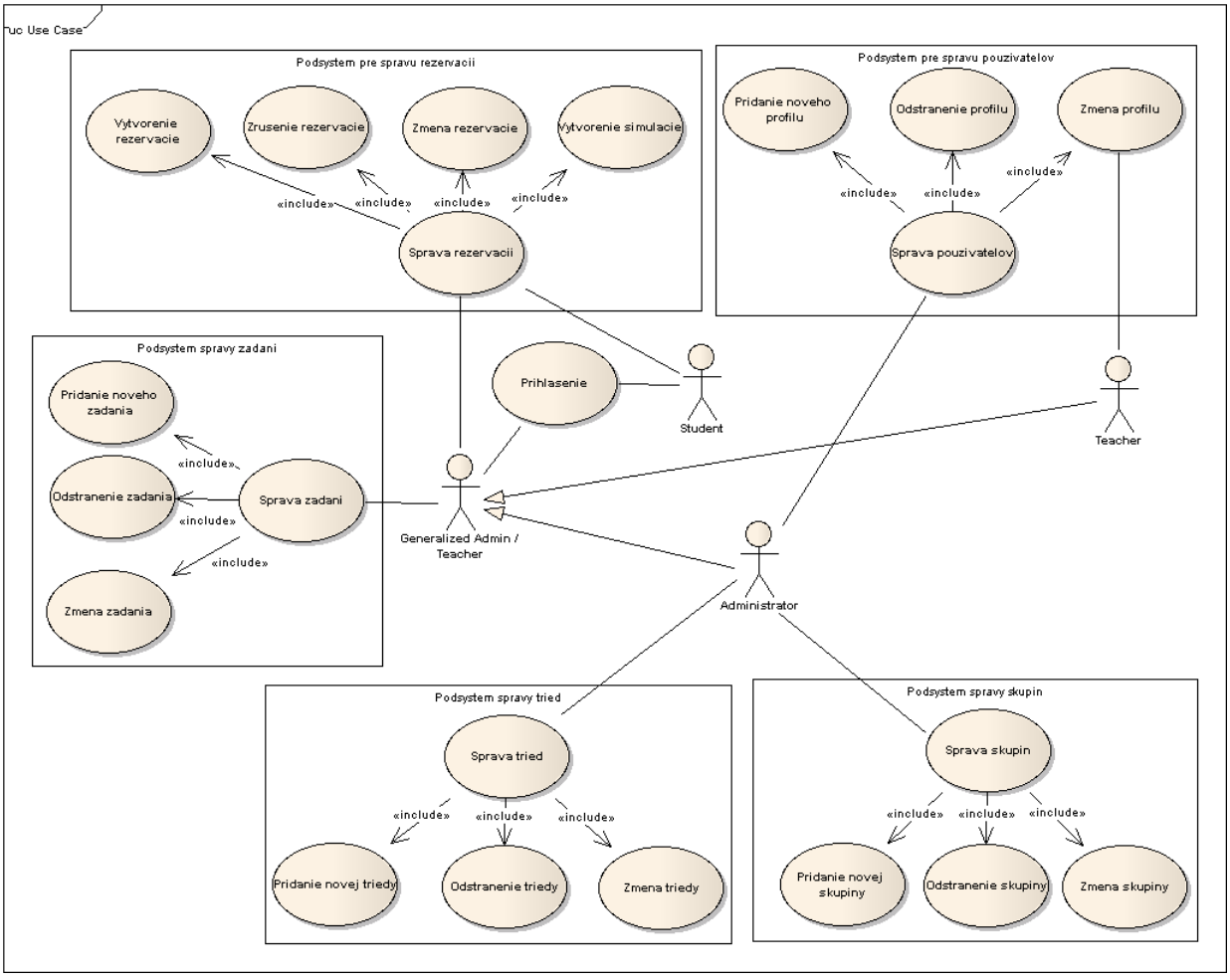
Novým záznamom v tejto tabuľke je len name.

name - názov topológie.

Tabuľky Level a Class ostávajú nezmenené.

5.3.2 Prípady použitia

Na obrázku 5.2 je znázornený diagram prípadov použitia. V zásade sa medzi hráčmi objavila nová skupina rovnako boli zjemnené jednotlivé akcie.



Obr. 5.2: Diagram prípadov použitia

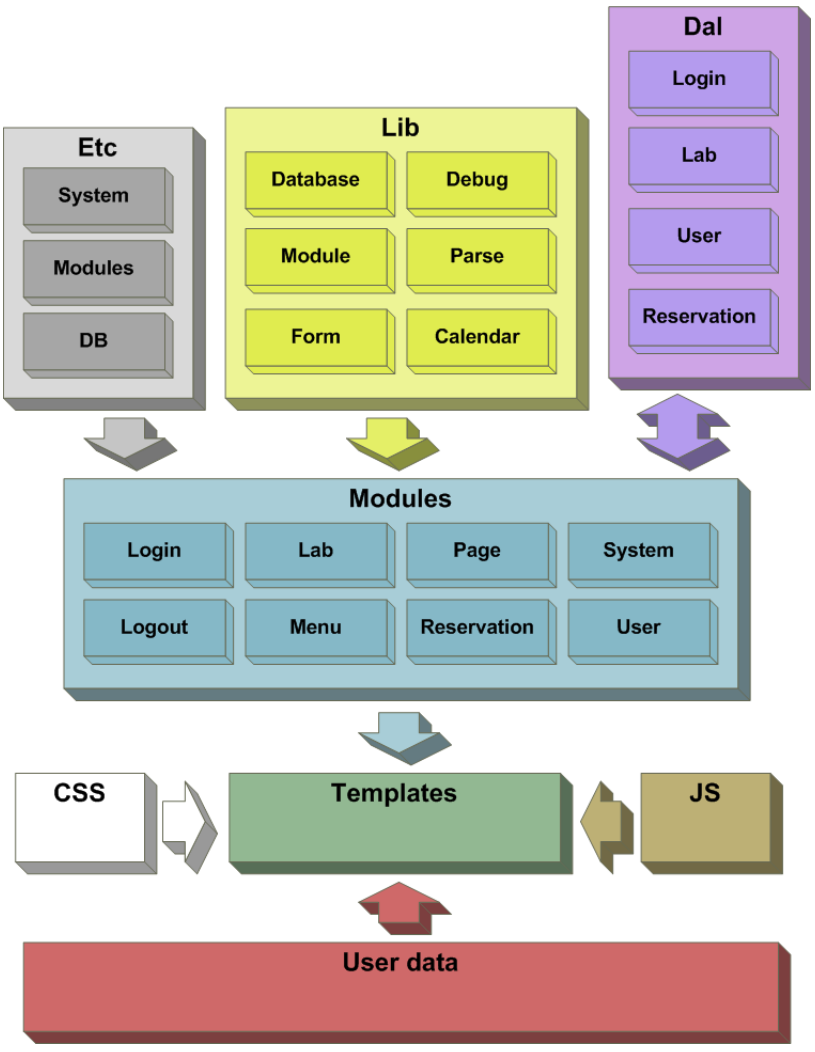
5.4 Web Interface

Pri systéme, ktorý sa skladá z viacerých častí, ktoré sú vzájomne spojené, je veľmi dôležité klásť dôraz na ich dobré logické usporiadanie. Tiež jednotný prístup pri písaní zdrojového

kódu veľmi uľahčuje prácu a zvyšuje prehladnosť. Z tohoho dôvodu sme vyžili objektovo orientované PHP. Organizácia do objektov umožňuje pomocou dedičnosti rozdeliť zdrojový kód na spoločné časti, ktoré môžu využívať viaceré moduly a špecifické časti, ktoré určujú správanie jednotlivých modulov.

5.4.1 Štruktúra

Systém, ktorým sa generujú stránky je možné rozdeliť na viacero logických častí. Preto sme tomu prispoôsobili aj štruktúru rozmiestnenia súborov. Tento prístup umožňuje ľahkú orientáciu a možnosť rýchlo modifikovať potrebné časti. Každá časť je uložená podľa jej významu (obr. 5.3).



Obr. 5.3: Organizácia web rozhrania

Moduly

Moduly sú v našom systéme jadrom, ktoré poskytuje používateľom rôzne funkcionality. Každý modul je rovnakým spôsobom vimplementovaný do systému a dodržuje určité spoločné konvencie. Modul môžeme rozložiť na viacero častí :

- Konfigurácia
- Knižnice
- Databázová vrstva
- Implementácia logiky

system
Vždy ako prvý sa vytvára modul system, ktorý tvorí základnú štruktúru vygenerovaných stránok. Ten ďalej určuje, ktorý modul sa bude volať. V prípade, že používateľ nie je prihlásený bude mu zobrazovať modul login až kým používateľ nevloží platné prihlasovanie údaje. Potom mu povolí prístup aj k ostatným modulom.

login
Modul zabezpečuje prihlásenie používateľa overením mena a hesla voči údajom v databáze. Pre zabezpečenie hesla sa do databázi ukladá iba hash vytvorený pomocou algoritmu SHA1. Pri vložení platných údajov modul nastaví v session premmenej, že daný používateľ je platne prihlásený.

Obr. 5.4: Výstup modulu login

logout
Tento modul po zavolaní zabezpečí, odhlásenie aktuálne prihlásený používateľ. K tomu je potrebné zmeniť nastavenie session premmých. Po ďalšom načítaní stránky systém bude opäť vyžadovať autentifikáciu.

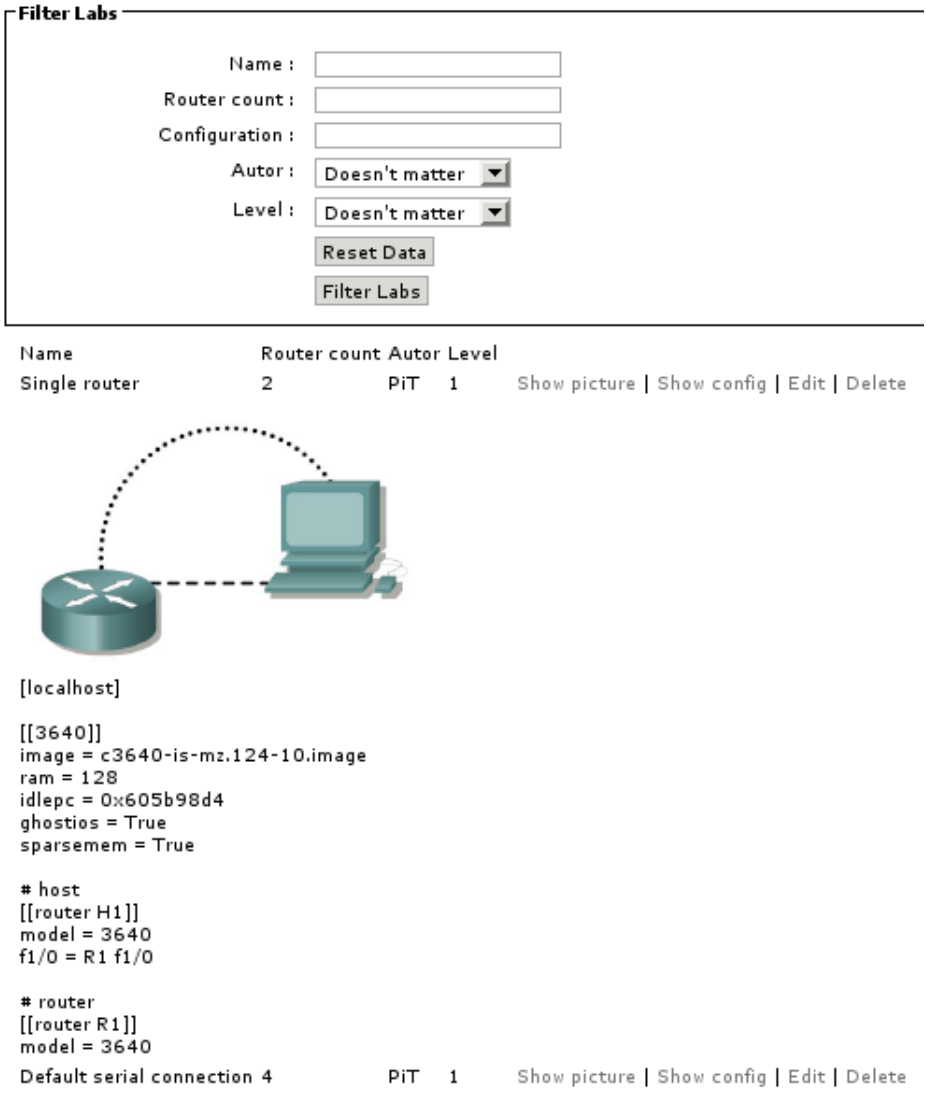
menu
Menu je jednoduchý modul, ktorý zobrazuje podľa konfigurácie a typu prihláseného používateľa odkazy na funkcie jednotlivých modulov.

page
Sa využíva na generovanie stránok s informáciami. V súčastnosti sa používajú iba statické stránky, ale časom bude rozšírený a obsah stránok bude generovaný z databázy.

Obr. 5.5: Výstup modulu menu

lab

Je modul na správu zadaní. Tu si môže oprávnený používateľ listovať všetky zadania v systéme. Tiež môže k nim pridávať nové, alebo editovať už existujúce, prípadne niektoré odstrániť.

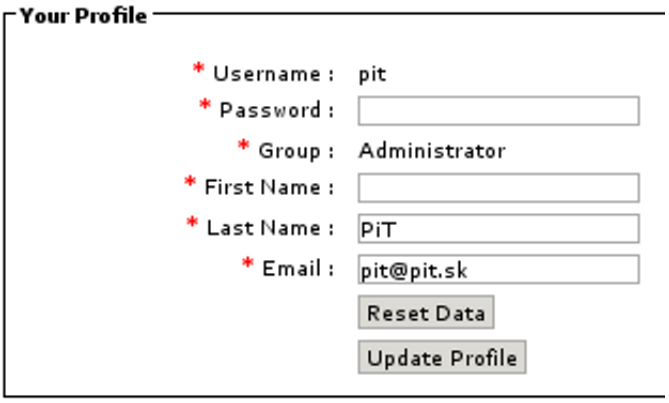


Obr. 5.6: Výstup modulu lab

user

Zabezpečuje správu používateľov. Používateľ s administrátorkým prístupom môže vytvárať

nové kontá, upravovať údaje jednotlivých používateľov, alebo odstranovať niektore znich. Tiež poskytuje možnosť obmedzene upravovať údaje aktuálne prihláseného používateľa.



Obr. 5.7: Výstup modulu user

reservation

Tento modul implementuje logiku vytvárania rezervácii. Ak je používateľ prihlásený v čase rezervácie, sprístupní sa mu možnosť výberu simulácie a následne pripojenia sa na konkrétne zariadenie.

Obr. 5.8: Výstup modulu reservation

Konfigurácia

Konfiguráciu si každý modul načítava sám. Je robená pomocou php skriptu, ktorý definuje pole s určitými parametrami, podľa ktorých sa bude daný modul správať. V súčastnosti využívame konfiguráciu pri nasledujúce časti systému :

- System - špecifikuje či sa má zobrazovať debug výpis z procesu vytvárania modulov. Toto je vhodné pri odlaďovaní systému, ale pred bežným používateľom chceme tieto informácie ukryť.

- Modules - je konfigurácia, ktorá sa používa pre každý modul a hovorí, ktoré úrovne používateľov možu používať jednotlivé funkcionality modulov. Je to jednoduchý mechanizmus zabezpečenia. Tiež sa podľa toho generujú položky v menu.

- Database - obsahuje prístupové údaje do databázy.

Takéto využitie konfiguračných skriptov nám prináša možnosť zmeniť parametre od ktorých sa bude odvýjať správanie celého systému.

Knižnice

Pomocou knižníc definujeme objekty, ktoré sa využívajú v rôznych častiach systému, aby sme sa vyhli opakovaniu rovnakého kódu. Tiež sa využívajú ako spoločný základ pred ďalším rozšírením.

debug

Sa používa vo všetkých moduloch a knižniciach ako jednotný spôsob pre výpis debug a error hlášok. Zjednotenie hlášok od všetkých modulov môžeme použiť ako vstup pre správu ak nastane chyba, aby sme vedeli jednoznačne určiť, čo ju spôsobilo. Tatiež sú hlášky v poradí a na rovnakom mieste, tak je jednoduché znich vyčítať čo systém robí.

database

Je objekt, ktorý zabezpečuje prístup k databáze. Poskytuje funkcie na pripojenie pomocou parametrov, ktoré nájde v konfigurácii. Zabezpečuje vykonávanie dotazov do databázy a spracovanie výsledkov, ktoré odovzdáva ako výstup. Tiež spracováva chyby počas vykonávania dotazov. Na začiatku modul systém pomocou tejto knižnice vytvorí spojenie k databáze, ktoré sa predávané ďalej všetkým ostaným modulom, aby nemusel každý vytvárať nové spojenie. Ďajel sa používa aj ako spoločný základ pri implementácii databázovej vrstvy jednotlivých modulov.

parser

Je rozhranie medzi vstupom do šablón a systémom Smarty. Sem moduly vkladajú svoje výstupy a následne určujú ktorou šablónou sa majú parsovať. Výsledný HTML kód sa dáva ako výstup z modulu, ktorý sa môže opäť vkladať do inej šablóny ako výstup z volaného modulu. Tento postup ne možné rekurzívne opakovať lubovolne veľa krát.

module

Implementuje všobecný modul, ktorý poskytuje spoločné funckie pre všetky moduly. Neskôr sú od neho odvodené všetky moduly systému. Medzi jeho funkcie patrí napríklad použitie knižnice parser-a, alebo presmerovanie na inú stránku.

form

Rozširuje funkcionalitu objektu QuickForm2 z balíka PEAR, ktorý využívame na generovanie formulárov. Tonto objekt je veľmi šikovný a umožňuje rýchle a spolahlivé vytvorenie formulára, so základnými poliami, ktoré musia spĺňať určité pravidlá. Pred ich spravoaním vykóna overenie či boli splnené podmienky, ako napríklad použitie iba určitých znakov, alebo povinnosť vyplniť určené polia. Naša knižnica rozširuje objekt QuickForm2, aby sme mohli použiť rôzne šablony pri zobrazovaní vygenerovaných formulárov.

calendar

Slúži na generovanie dní v kalendári. Táto knižnica sa využíva pri rezervačnom systéme. Implementuje funkcie na prácu s časovými známkami (timestamp), ktoré sa používaju na uchovávanie času a dátumu. Pri genreovaní dní v mesiaci využíva spoločný prístup pre prácu so šablónami.

Databázová vrstva

Tieto objekty defunujú všetky operácie nad databázou, ktoré sa používajú v danom module. Každý modul si vytvorí inštanciu vlastnej databázovej vrstvy a ako parameter jej vloží spojenie na databázu, ktoré obdržal od nadradeného modulu, ktorý ho vytvoril. Potom už môže volať funkcie, ktoré mu vrátia pole s požadovanými údajmi z databázy. Sú tu taktiež definované všetky dotazy, aby v samotných moduloch zostala úž iba logika, ktorá si údaje vyžiada a následne ich spracuje. Týmto sme oddelil všetku prácu s databázou a prehladne ju sprístupnili cez jeden objekt. Taktiež sme zjednotili správanie pri chybách.

Šablóny

Sú textové súbory, ktoré obsahujú všetok HTML kód a špeciálne značky pre systém Smarty. Ten podla nich viem kam má ktoré údaje nahradiť. Tiež sú tu zapísane jednoduché cykli a podmienky na dynamické generovanie stránok. Keď Smarty preparsuje celý súbor odstráni všetky bloky, ktoré boli učené jemu a vráti čistý HTML kód s vloženými údajmi. Týmto spôsobom oddelíme HTML kód (takzvanú prezentačnú vrstvu) od vnútornej logiky. Keď potrebujeme upraviť výstup modulu všetko nájdeme v šablóne a nemusíme prechádzať PHP kód, ako je tomu v systémoch, ktoré nepoužívajú podobný spôsob generovania výstupov.

```
<table>
{ foreach from=$day key=hour item=hour_details }
    <tr width="50px">
        { if $hour_details.available AND $hour_details.timestamp > $now }
            { if $hour_start_timestamp }
                <td><a href="index.php?mod=reservation&act=new&
                   hour_start={ $hour_start_timestamp }&
                   hour_end={ $hour_details.timestamp }">{ $hour }:00</a></td>
                   <td>This hour is available</td>
            { else }
                <td><a href="index.php?mod=reservation&act=new&
```

```
                hour={ $hour_details.timestamp }">{ $hour }:00</a></td>
                <td>This hour is available</td>
            { /if}
    { else }
        <td>{ $hour }:00</td><td>This hour is not available</td>
    { /if }
     </tr>
{ /foreach }
</table>
```

Tu môžeme vidieť šablónu pre kalendár. Všetky bloky ohraničené znakmi { } sú určené pre systém Smarty. Syntax je veľmi podobná jazyku PHP.

Pre úpravu vzhľadu využívame caskádové štýly CSS. Tiež je tu snaha maximálne oddeliť štýly od HTML kódu.

Ďalším rozšírením sú Javaskripty, ktoré majú v systéme tiež osobitné miesto.

Celkový výstup dopĺňajú údaje ako obrázky, alebo objekty, ktoré sa vkladajú do HTML kódu. Tie sú uložené vo vlastnom adresári.

5.4.2 Objekty web rozhrania

Pre implementáciu sme si navrhli triedy a ich vzájomnú dedičnosť, ako sú zobrazené v prilohe 1. Správna analýza tried nám pomáha, aby sme sa vyhli duplicitným kódom, keď všetky spoločné funcie sú implementované v triede rodiča a triedou potomka doimplentúvame iba špecifické funckie danej triedy. Môžeme to vidieť napríklad pri implementácii databázovej vrstvy, kde naprv implemtujeme triedu pre všeobecnú prácu s databzou. Neskôr od nej odvodzujeme triedy so špecifickými dotazmi pre daný modul. Týmto spôsobom vieme sprehľadniť kód a veľmi zjednodušiť úpravy. Ak sa vyskytne chyba v implementácii niektorej funckie stačí ju opraviť na jednom mieste aj keď sa používa v desiatkach iných modulov. Objektovo orientovaný prístup nám veľmi uhľahčil orientáciu v kóde a dodal mu logickú štruktúru.

5.5 Proxy

Architektúra

Program je viacvláknový, pričom každé príchodzie spojenie je spracované samostatným vláknom. Program sa spúšťa ako jeden proces, ktorý načúva na zvolenom TCP porte a v prípade prijatia spojenia vytvára nové vlákno pre spracovanie tohto spojenia.

Najprv sa vedie úvodný dialóg s telnet appletom a v prípade úspešnej autentifikácie je otvorené spojenie na cieľový port virtuálneho smerovača.

Komunikácia s klientom

Program čaká na zvolenom TCP porte na vonkajšom sieťovom rozhraní systému na spojenia iniciované Java Telnet Appletom. Klientovi je zobrazený úvodný pozrav.

```
Proxy vlab at your service
```

Ako odpoveď sa od klienta očakáva správa v nasledujúcom formáte:

```
relay auth port
```

Kde 'relay' je konštantný reťazec, 'auth' je autentifikačný reťazec vygenerovaný web aplikáciou a 'port' je číslo portu na ktorý sa klient pokúša pripojiť. Správa je ukončená znakom nového riadku. V prípade nedodržania formátu je do spojenia vypísaná chybová správa a spojenie ihneď ukončené.

V prípade úspešnej autentifikácie sú všetky nasledujúce dáta preposielajú na cieľový port.

Autentifikácia

Autentifikácia klienta spočíva v kontrole správnosti 'auth' reťazca a portu. Ak súhlasí reťazec so záznamom pre danú simuláciu a zároveň je požadovaný port platným konfiguračným portom prislúchajúcej simulácie, je povolený prístup k tomuto portu.

Komunikácia s databázou

Správnosť uvedených informácií sa overuje dotazom do databázy.

Dotaz má formát:

```
1:  SELECT  s.auth_hash,s.id_lab,s.first_port,l.router_count,l.host_count \
    FROM simulation s,lab l WHERE\
2:  s.auth_hash = 'auth_string' AND
3:  s.id_lab = l.id AND\
4:  port_number  > s.first_port AND \
    port_number  <= (s.first_port + l.router_count + l.host_count + 1)
```

Prvý riadok je výberom potrebných stĺpcov, druhý riadok overuje správnosť autentikačného reťazca, tretí riadok kontroluje, či reťazec prislúcha danej simulácii a posledný riadok overuje, či požadovaný port patrí k aktuálnej simuláci. V dotaze sa s jednotlivými pripojeniami mení iba položka 'auth_string', ktorá zodpovedá reťazcu zadanému klientom, a 'port_number', ktorá je taktiež klientom požadovaným cieľovým portom.

Ak boli zadané správne údaje, výstupom dotazu je riadok záznamu v databáze. Ak boli zadané nesprávne údaje, daný záznam nebude nájdený a odpoveď na dotaz bude prázdna, alebo bude v prípade poruchy obsahovať chybovú správu.

Formát denníka

Proxy program vytvára kvôli kontrole prístupu a voliteľne pre ladiace účely denník.

```
ctime      cas      datum      ip adresa     port1 port2 auth  sprava
1209905577 12:52.57 04.05.2008 Vlab proxy started
1209905588 12:53.08 04.05.2008 78.99.166.153 35182 22 password CONNECT
1209905589 12:53.09 04.05.2008 78.99.166.153 35182 22 password DISCONNECT
1209905982 12:59.42 04.05.2008 78.99.166.153 35040 30 password ERR_CLIENT
```

Formát denníka je nasledovný: Ak nejde o správu o naštartovaní servera, prvou položkou je dátum a čas najprv v podobe sekúnd pre počítačové spracovanie, následne v čitateľnej podobe, potom IP adresa a port z ktorého sa pripája klient, port na ktorý sa klient snaží pripojiť, autentifikačný reťazec ktorý použil a správa.

Možné správy sú:

CONNECT klient sa úspešne autentifikoval a pripojil na žiadaný port

DISCONNECT spojenie s klientom bolo úspešne ukončené

ERR_CLIENT nepodarilo sa naviazať spojenie na klientskej časti

ERR_INIT klient zadal nesprávny vstupný reťazec, pravdepodobne cudzí program

ERR_AUTH klientovi sa nepodarilo autentifikovať

Kapitola 6

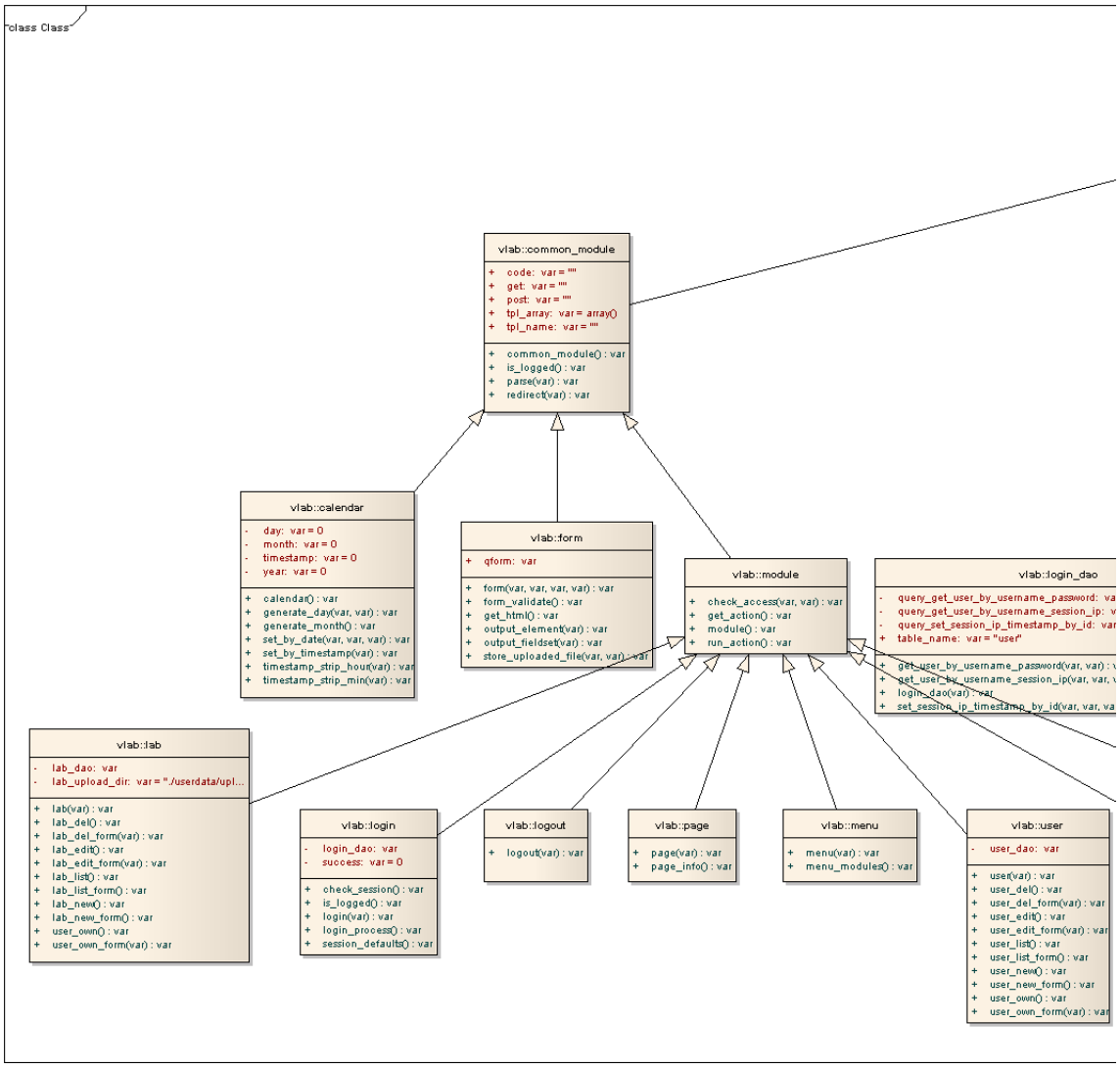
Zhodnotenie

V priebehu letného semestra sa aj v spolupráci s vedúcim projektu vynorilo množstvo zaujímavých možností, ktoré by systém mohol obsahovať. Návrh systému ako celku si však

žiada viac času na implementáciu ako sme predpokladali. Z toho dôvodu bolo nutné niektoré časti vypustiť, prípadne zredukovať. Atraktívna téma na druhej strane jedného člena tímu, konkrétne Petra Pétiho zaujala natoľko, že sa rozhodol v nej pokračovať aj v rámci diplomového projektu.

Počas týchto dvoch semestrov sme sa naučili, aké je to pracovať v tíme, kedy človek nerozhoduje o všetkom sám, ale všetci participujú na všetkom. Zistili sme, aké úskalia práca v tíme prináša, ako môže dobrá tímová komunikácia urýchliť vývoj produktu a predovšetkým, že jedine koordinovaná tímová práca môže priniesť želané výsledky.

Príloha 1: Logický model databázy



Príloha 1: Logický model databázy