Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Interaktivní kurz o internetových protokolech pro střední školy

Martin Greger

Vedoucí práce: Ing. Ivan Ryant

7. května 2015

Poděkování

Rád bych poděkoval svému vedoucímu práce Ing. Ivanu Ryantovi za vedení práce a své rodině za podporu během studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 7. května 2015 .....

Odkaz na tuto práci

Greger, Martin. *Interaktivní kurz o internetových protokolech pro střední školy.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Práce se zabývá tvorbou interaktivního vzdělávacího kurzu o internetových protokolech pro střední školy. Kurz vysvětluje základní principy protokolů TCP, UDP a IP. Práce popisuje analýzu, návrh a realizaci tohoto kurzu.

Klíčová slova e-learning, vzdělávací kurz, síťové protokoly, TCP, UDP, IP

Abstract

Thesis deals with creation of an interactive educational course about internet protocols for high schools. The course explains basic principles of protocols TCP, UDP and IP. Thesis describes analysis, design and realization of this course.

Keywords e-learning, educational course, network protocols, TCP, UDP, IP

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V současné době je více a více podporováno zapojení informačních technologií do běžné výuky. Avšak počítačů a podobných zařízení je na středních školách již dostatek, stále chybí dostatečně kvalitní software, který výuku podporuje. Systémy typu Moodle pro střední školy nejsou nejvhodnějším řešením, protože jsou poněkud náročnější na provoz a administraci. Vhodným příkladem softwaru podporující výuku je interaktivní kurz. Interaktivní kurz by měl podporovat celý proces výuky od vysvětlení látky, po její procvičení a následné otestování získaných vědomostí. Oblastí, kterých lze pomocí interaktivního kurzu podporovat, je mnoho, avšak pro tuto práci byla vybrána oblast internetových protokolů. Internetových protokolů je velké množství, proto byly pro tento kurz vybrány jen ty základní a nejčastěji používané, o kterých by student střední školy měl mít povědomí. Jedná se o protokoly TCP, UDP a IP. Použitím tohoto kurzu kantor efektivně naučí studenty znalostem problematiky daných protokolů a ušetří drahocený čas během výuky. Dále má možnost znalosti studentů otestovat a tím i ušetřit čas spojený s přípravou testu. Tato práce vzniká za účelem podpory a usnadnění výuky internetových protokolů na středních školách.

Kapitola 1

Cíl práce

Cílem této práce je vytvořit interaktivní e-learningový kurz o internetových protokolech. Tento systém má za úkol seznámit středoškoláky s vybranými internetovými protokoly.

1.1 Požadavky

1.1.1 Funkční požadavky

- kurz bude obsahovat stručný úvod do modelu TCP/IP
- kurz bude obsahovat následující protokoly: TCP, UDP, IP
- kurz bude pro každý protokol obsahovat tyto sekce: kurz, cvičení a test

1.1.2 Nefunkční požadavky

- jednoduché ovládání, práci s aplikací by měli být schopni i méně zdatní studenti a kantoři
- jednoduchá instalace a konfigurace systému vzhledem k technickým dovednostem kantora
- programovací jazyk Java, tento požadavek zahrnut již v zadání této práce

1.2 Užití

Používání kurzu je vcelku jednoduché. Pro každý z protokolů je v hlavním okně stručný popis toho, co je obsahem kurzu. U každého protokolu má uživatel na výběr ze tří sekcí: kurz, cvičení a test. Tyto sekce jsou popsány níže. Po kliknutí na tlačítko s vybranou sekcí se uživateli zobrazí okno s kurzem/cvičením/testem. Uživatel může pomocí tlačítek Další/Předchozí listovat obsahem

vybrané sekce. Mimo toho, kurz také obsahuje stručný úvod do TCP/IP modelu, který však obsahuje pouze možnost kurzu. Úvod usnadňuje pochopení fungování samotných protokolů. Podrobnější popis použití a instalace kurzu je popsán v uživatelské dokumentaci, která je obsahem přiloženého CD.

1.2.1 Kurz

V sekci kurz jsou vždy vysvětleny základní principy fungování konkrétního protokolu. Principy jsou vysvětleny pomocí animací, které si uživatel může přehrát a také obsahuje interaktivní strukturu datagramu. Po najetí myší na vybranou položku datagramu kurz zobrazí základní informace o této položce.

1.2.2 Cvičení

Sekce cvičení je určena pro bližší pochopení fungování konkrétního protokolu. Obsahem této sekce jsou animace, které si uživatel může v rámci možností sám konfigurovat a přehrát. Princip konfigurovatelných animací umožňuje bližší pochopení fungování protokolu a vyjasňuje případné nejasnosti. Dále má uživatel možnost se seznámit s podrobnější strukturou datagramu prostřednictvím interaktivní struktury datagramu jako v kurzu (informace o konkrétní položce jsou však podrobnější než v sekci kurz).

1.2.3 Test

Sekce test je určena zejména pro otestování znalostí o konkrétním protokolu. Je tvořena sadou otázek. Otázky jsou trojího typu: otázky s radiobuttony, otázky s checkboxy a otázky s textovou odpovědí. Otázka s radiobuttony a textová otázka má vždy jednu správnou odpověď, otázka s checkboxy má počet správných odpovědí libovolný. Po vyplnění otázek má uživatel možnost uložení souboru s výsledky.

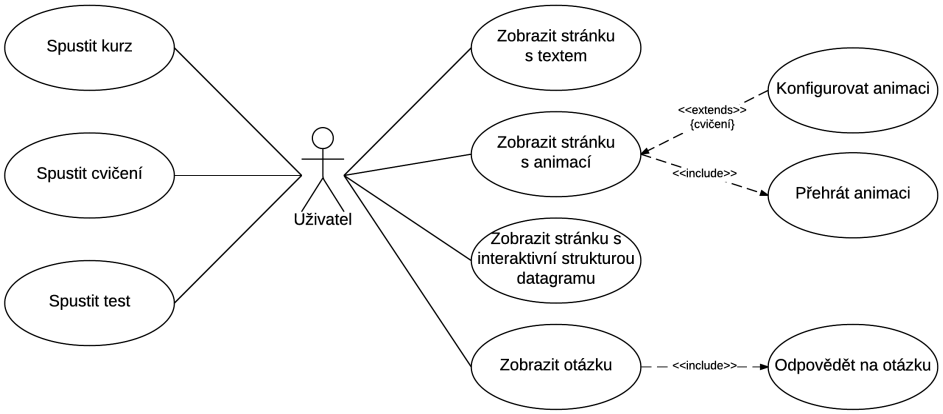
1.3 Případy užití

Tato sekce popisuje nejčastější případy užití kurzu. Diagram případů užití je zobrazen na obrázku 1.1.

1.3.1 Průběh testu

Hlavní scénář

1. uživatel klikne na tlačítko Test
2. systém vygeneruje a zobrazí otázky
3. uživatel vyplní odpovědi na otázky



Obrázek 1.1: Případy užití

4. uživatel klikne na tlačítko Ukončit a ohodnotit
5. systém ohodnotí test, v případě otázek s radiobuttony a checkboxy zobrazí správné odpovědi
6. uživatel klikne na tlačítko Uložit výsledek
7. systém zobrazí dialog s umístěným souboru s výsledky
8. uživatel vybere místo v adresáři a potvrdí ho
9. systém uloží soubor s výsledky testu
10. uživatel vypne okno s testem

Alternativní scénář

1. tento scénář navazuje na 7. bod hlavního scénáře
2. uživatel stornuje uložení výsledku
3. systém zruší dialog s umístěným souboru s výsledky
4. uživatel vypne okno s testem

1.3.2 Zobrazení další/předchozí stránky s obsahem

Hlavní scénář

1. uživatel klikne na tlačítko Další/Předchozí
2. systém zobrazí další/předchozí stránku s obsahem (text/animace/obrázek/otázka)

Alternativní scénář

1. tento scénář navazuje na 1. bod hlavního scénáře
2. systém zobrazí první/poslední stránku
   s obsahem (text/animace/obrázek/otázka)
   a deaktivuje tlačítko Další/Předchozí

1.3.3 Zobrazení a konfigurace animace

Hlavní scénář

1. uživatel nastaví parametry animace
2. uživatel spustí animaci
3. systém přehraje animaci s nastavenými parametry

Alternativní scénář

1. uživatel spustí animaci
2. systém přehraje animaci se základní konfigurací

1.3.4 Zobrazení interaktivní struktury datagramu

Hlavní scénář

1. uživatel najede myší na položku datagramu
2. systém zobrazí informace o vybrané položce

Kapitola 2

Přehled existujících řešení

Srovnání vybraných existujících řešení je zobrazeno v tabulce 2.1. Vybrané e-learningové systémy jsou popsány níže.

2.1 Moodle

Moodle je softwarový balíček pro tvorbu výukových systémů a elektronických kurzů na internetu. Jedná se o neustále se vyvíjející projekt, navržený na základě sociálně konstruktivistického přístupu k vzdělávání [1].

2.1.1 Výhody

Systém Moodle poskytuje téměř neomezené možnosti při tvorbě studiních materiálů. Jedná se nejen o kurzy, ale i mechanizmy k procvičení probírané látky a k následnému otestování znalostí studentů. Další výhodou tohoto systému je fakt, že Moodle je považován za synonymum pro e-learningový systém a většina škol, který se pro využití nějakého e-learningového systému rozhodnou, zvolí právě tento systém. Mezi výhody tohoto systému je i fakt, že je publikován pod licencí open-source.

2.1.2 Nevýhody

Hlavní nevýhodou systému Moodle pro použití pro střední školy jsou jeho požadavky. Vzdělávací systém Moodle potřebuje:

- běžící databázový server
- software pro webový server
- běžící PHP

To pro většinu středních škol představuje problém, protože nemají k dispozici požadované technologie k úspěšnému nasazení tohoto systému.

2.2 Claroline

Claroline je alternativou k systému Moodle. Je vyvíjen převážně francouzskými a belgickými univerzitami, kde se také používá nejvíce.

2.2.1 Výhody

Výhodou tohoto systému fakt, že je publikován pod licencí open-source.

2.2.2 Nevýhody

Hlavní nevýhodou tohoto systému je nekompletní česká lokalizace (něco přeloženo je, něco ne) a podobná náročnost na technologie jako Moodle. Dále je možné zmínit omezené množství rozšiřujících modulů v porovnání se systémem Moodle.

2.3 Netventic Learnis

Jedná se o komerční vzdělávací systém, který je vyvíjen českou firmou Netventic Technologies. Tento systém je převážně určen pro české a slovenské školy.

2.3.1 Výhody

Výhodou tohoto systému je, že je vhodný pro všechny typy škol. Další výhodou tohoto systému je, že obsahuje plnohodnotnou českou lokalizaci. Dále sem lze zahrnout možnost školení pro kantory, online podporu a další služby spojené s provozem a užitím tohoto systému.

2.3.2 Nevýhody

Hlavní nevýhodou tohoto systému je, že provoz i některé z výše uvedených služeb jsou placené.

Tabulka 2.1: Srovnání e-learningových systémů

Systém	Provoz	Čestina	Potřebné technologie
Moodle	zdarma	ano	webový server databázový server PHP
Claroline	zdarma	ne	webový server MySQL databázový server PHP
Netventic Learnis	placený	ano	dodané při nasazení dodavatelem

Kapitola 3

Analýza protokolů

Obsahem této kapitoly je analýza protokolů zobrazených v kurzu a také popis TCP/IP modelu, který je do kurzu také zahrnut. U každého protolu jsou popsány jeho vlastnosti, struktura datagramu, základní princip fungování a způsob, jakým je protokol zobrazen v kurzu. Dále zde najdeme popis a základní vlastnosti modelu TCP/IP.

3.1 Seznam pojmů

3.1.1 Datagram

Datagram je pojem označující základní datovou jednotku na dané vrstvě modelu TCP/IP. Jedná se o přenášená data spolu s hlavičkou (případně i patičkou) protokolu konkrétní vrstvy.

3.1.2 Záhlaví datagramu

Záhlavím datagramu jsou myšleny všechny položky datagramu kromě přenášených dat.

3.1.3 Segment

Segment je pojem označující datagram v protokolu TCP.

3.1.4 Paket

Paket je pojem označující datagram v protokolu IP.

3.1.5 Rámec

Rámec je pojem označující datagram ve vrstvě síťového rozhraní.

3.1.6 Spolehlivý protokol

Spolehlivý protokol je takový protokol, který při případné ztrátě konkrétního datagramu zajistí jeho opětovné odeslání. Spolehlivý protokol je například protokol TCP.

3.1.7 Nespolehlivý protokol

Nespolehlivý protokol je takový protokol, který se při případné ztrátě konkrétního datagramu nezabývá jeho znovuodesláním. Nespolehlivý protokol je například protokol UDP.

3.1.8 Spojovaný protokol

Spojovaný protokol je takový protokol, který pro přenos dat vyžaduje vytvořit spojení, po celou dobu přenosu dat spojení udržovat a na konci přenosu toto spojení ukončit. Spojovaný protokol je například protokol TCP.

3.1.9 Nespojovaný protokol

Nespojovaný protokol je takový protokol, který pro přenos dat žádné spojení nevyžaduje. Nespojovaný protokol je například protokol UDP.

3.1.10 Stavový protokol

Stavový protokol je takový protokol, který se k odesílaným nebo přijatým datagramům chová podle stavu, ve kterém se nachází.

3.1.11 Bezstavový protokol

Bezstavový protokol je takový protokol, který se k odesílaným nebo přijatým datagramům chová pořád stejným způsobem.

3.1.12 MTU

MTU je zkratka pro maximum transmission unit. Maximum transmission unit udává maximální možnou velikost datagramu, který lze přes dané síťové rozhraní poslat.

3.2 TCP/IP model

Tato sekce obsahuje základní informace a principy modelu TCP/IP. Informace a principy jsou čerpány ze zdrojů [2, 3].
Jedná se o síťovou architekturu, která popisuje průběh komunikace mezi jednotlivými zařízeními v síti. TCP/IP model dělí síťovou komunikaci do čtyř

vrstev (narozdíl od modelu ISO/OSI, který síťovou komunikaci rozděluje do sedmi vrstev). Patří sem vrstva aplikační, transportní, síťová a o vrstva síťového rozhraní.

3.2.1 Aplikační vrstva

Jedná se o vrstvu aplikací, které používají pro přenos dat síť. Lze sem zařadit například elektronickou poštu, sdílení souborů nebo vzdálený přenos dat. Mezi protokoly této vrstvy patří například: HTTP, HTTPS, FTP, SMTP, POP3, SSH atd.

3.2.2 Transportní vrstva

Úkolem transportní vrstvy je zajišťení komunikace po síti pro aplikace z aplikační vrstvy. Transportní vrstva pak předává data přímo aplikacím, které si tyto data vyžádaly (narozdíl od vrstev nižších). Mezi protokoly této vrstvy patří TCP a UDP. Existují i další protokoly, které lze do transportní vrstvy řadit (např. protokol DCCP), avšak převážně jsou používány již zmíněné protokoly TCP a UDP. Je pak na konkrétní aplikaci, zda pro přenos dat použije rychlý, avšak nespojovaný a nespolehlivý protokol UDP, nebo na režii náročnější, spojovaný a spolehlivý protokol TCP.

3.2.3 Síťová vrstva

Síťová vrstva má na starost hledání cesty mezi libovolnými uzly sítě a následné doručení paketů. Při hledání cesty sítí se bere ohled hlavně na rychlost přenosu dat, na spolehlivost nikoliv. Mezi protokoly této vrstvy patří: IP, ICMP, ARP atd.

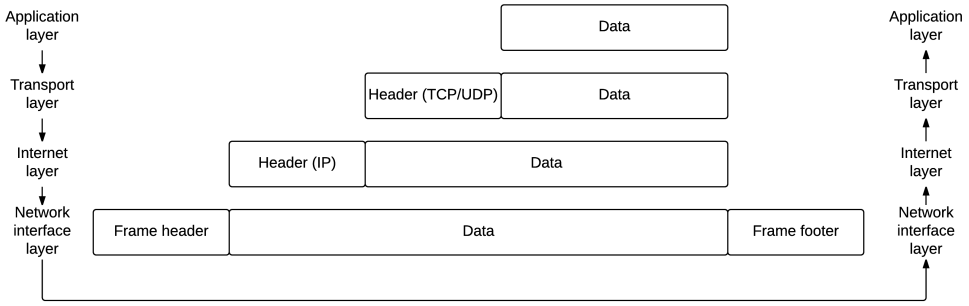
3.2.4 Vrstva síťového rozhraní

Vrstva síťového rozhraní umožňuje přístup k přenosovému médiu. Úkolem této vrstvy je přenos rámců mezi sousedními uzly v síti. Tato vrstva není v TCP/IP modelu konkrétněji specifikována, protože je závislá na své implementaci.

3.2.5 Encapsulace dat

Při odesílání dat do sítě se na každé z vrstev TCP/IP modelu k datům přidává hlavička (případně i patička) protokolu dané vrstvy. Tento proces se nazývá encapsulace dat a probíhá následujícím způsobem.

1. Aplikační vrstva předá odesílaný blok dat transportní vrstvě.
2. Transportní vrstva přijme tento blok dat a přidá před něj TCP nebo UDP hlavičku, vzniklý TCP segment nebo UDP datagram je předán síťové vrstvě.



Obrázek 3.1: Encapsulace a deencapsulace v modelu TCP/IP

3. Síťová vrstva přijme tento TCP segment nebo UDP datagram, přidá před něj IP hlavičku a vzniklý IP paket je předán vstvě síťového rozhraní.

4. Vrstva síťového rozhraní přijme tento IP paket, vytvoří rámec přidáním hlavičky před přijatý paket a patičky za přijatý paket a rámec je odeslán do sítě.

3.2.6 Deencapsulace dat

Jedná se o proces encapsulace v opačném pořadí. V každé z vrstev je z přijatého bloku dat odebírána hlavička (případně i patička) protokolu dané vrstvy. Proces encapsulace a deencapsulace dat je popsán na obrázku 3.1.

3.2.7 Způsob zobrazení modelu v kurzu

V úvodu kurzu je vysvětlen základní princip modelu TCP/IP. Jsou zde popsány vrstvy tohoto modelu a také princip zapouzdření dat při síťové komunikaci. Proces zapoudření dat je doplněn o animaci, která tento princip vysvětluje.

3.3 IP protokol

Tato sekce obsahuje základní informace a principy protokolu IP. Informace a principy jsou čerpány ze zdrojů [4, 5] pro IP protokol verze 4, ze zdrojů [6, 7] pro IP protokol verze 6.
IP protokol je nespolehlivý protokol síťové vrstvy (případně požadovanou spolehlivost musí zaručit protokol vyšší vrstvy). Úkolem IP protokolu je adresace a směrování paketů. Dále se stará o fragmentaci paketů (není vždy nutná).

Tabulka 3.1: Zápis IPv6 adresy

Typ zápisu	IP adresa
Základní	2001:0db8:0000:0000:0000:0000:1428:57ab
Zkrácená dvojtečková notace	2001:0db8:0000:0000:0000::1428:57ab
Poslední 4B desítkově s vynecháním nul	2001:0db8::20.40.87.171

3.3.1 IP adresa

Každé zařízení v síti(používající IP protokol) musí mít IP adresu. IP adresa je jednoznačný identifikátor konkrétního zařízení.

3.3.1.1 IP adresa verze 4

IP adresa verze 4 je 32 bitové číslo. Existuje 2^{32} takových adres. IP adresa verze se 4 se zapisuje v desítové soustavě po jednotlivých bajtech, které jsou od sebe odděleny tečkou.

3.3.1.2 IP adresa verze 6

IP adresa verze 6 je 128 bitové číslo. Existuje 2^{128} takových adres. Nejčastěji se IP adresa verze 6 zapisuje v hexadecimální soustavě po dvou bajtech, které jsou od sebe odděleny dvojtečkou. Různé zápisy IP adresy verze 6 jsou popsány v tabulce 3.1

3.3.2 Struktura paketu

3.3.2.1 Struktura paketu IPv4

Version
: je verze IP protokolu, v tomto případě se hodnota rovná 4

Internet header length
: je délka záhlaví IP paketu

Type of service
: je položka, která v praxi nenašla svého naplnění

Total length
: je délka celého IP paketu

Identification
: je identifikace IP paketu

Flags
: je pole příznaků

DF
příznak DF (don't fragment) se používá při fragmentaci, určuje, zda se paket může fragmentovat

MF
příznak MF (more fragments) se používá při fragmentaci, určuje, zda se jedná o poslední fragment

Fragment offset
položka používající se při fragmentaci, určuje počet bajtů, které byly vloženy předchozích fragmetů

Time to live
určuje dobu života paketu, zabraňuje případnému zacyklení paketu v síti

Protocol
je číselná identifikace protokolu vyšší vrstvy

Header checksum
je kontrolní součet ze záhlaví IP paketu

Source address
je IP adresa odesílatele

Destination address
je IP adresa příjemce

Options
jsou volitelné položky, používány ojediněle

Data
jsou přenášená data

3.3.2.2 Struktura základní hlavičky IPv6

Version
je verze IP protokolu, v tomto případě se hodnota rovná 6

Traffic class
při zahlcení sítě specifikuje, která data mohou být zahozena dříve a která později

Flow label
spolu s adresou odesilatele jednoznačně identifikuje jeden dílčí tok dat v síti

Payload length
je délka IP paketu bez základního záhlaví

Tabulka 3.2: Další hlavičky v IPv6

Číslo hlavičky	Typ hlavičky
0	Hop-by-Hop Header
4	IP protocol
6	TCP protocol
17	UDP protocol
43	Routing header
44	Fragment header
45	Protocol IRP
46	Protocol RRP
50	Encrypted security payload header
51	Authentication header
58	Protocol ICMP
59	No next header
60	Destination options header

Next header
je typ následující hlavičky

Hop limit
odpovídá položce Time to live IP paketu verze 4

3.3.2.3 Další hlavičky IPv6

Položka Next header IP paketu verze 6 určuje typ následující hlavičky. Může ukazovat na rozšiřující hlavičky protokolu IPv6, nebo na protokol vyšší vrstvy. Typy dalších hlaviček jsou popsány v tabulce 3.2.

3.3.3 Fragmentace

Fragmentace IP paketu je proces, při kterém je IP paket rozdělen na menší části (fragmenty). Proces fragmentace se používá v případě, když je nutné po přenosové lince přenést větší paket než je MTU linky.

3.3.3.1 Fragmentace IPv4

Fragmentaci IP paketu verze 4 může provádět jakékoliv zařízení v síti. Fragmentace IP paketu verze 4 probíhá následujícím způsobem:

1. Směrovač přijme paket, který je větší než MTU následující linky.

2. Paket je rozdělen na fragmenty, každému fragmentu jsou
v hlavičce (převzaté z původního paketu) nastaveny položky Fragment

Tabulka 3.3: Směrovací tabulka

Cíl v síti	Maska	Brána	Rozhraní	Metrika
0.0.0.0	0.0.0.0	192.168.1.1	192.168.1.16	25
127.0.0.0	255.0.0.0	On-line	127.0.0.1	306
192.168.1.16	255.255.255.255	On-line	192.168.1.16	281

offset, Total length a příznak More fragments (hodnota 0 určuje poslední fragment, hodnota 1 určuje, že fragment poslední není). Položka Identification je pro všechny fragmenty stejná jako v původním paketu.

3. Fragmenty jsou odeslány.

3.3.3.2 Fragmentace IPv6

Fragmentaci IP paketu verze 6 může pouze odesílatel. Fragmentace IP paketu verze 6 probíhá následujícím způsobem:

1. Směrovač přijme paket, který je větší než MTU následující linky.
2. Paket je zahozen a je odeslána chybová hláška pomocí protokolu ICMP odesílateli.
3. Odesílatel provede fragmentaci, za základní hlavičku (převzaté z původního paketu) každého fragmentu je přiřazena Fragment header.
4. Fragmenty jsou odeslány.

3.3.4 Směrovací tabulka

Směrovací tabulkou je myšlena datová struktura v operační paměti zařízení v síti (počítač, směrovač atd.), která obsahuje informace potřebné ke směrování. Příklad směrovací tabulky je zobrazen v tabulce 3.3.

3.3.5 Přenos

1. Odesílatel dostane data od transportní vrstvy.
2. Odesílatel otestuje, zda se nachází ve stejné síti jako příjemce.
3. Pokud ano, dochází k přímému doručování paketů (nejedná se o směrování).
   a) Pokud je adresa odesílatele shodná s adresou příjemce, data jsou předána transportní vrstvě.
   b) Pokud je adresa odesílatele jiná, paket je předán vrstvě síťového rozhraní k doručení.

4. Pokud ne, dochází ke směrování.
5. Odesílatel prohledá směrovací tabulku.
   a) Pokud je v tabulce záznam o trase k příjemci, paket je odeslán touto trasou.
   b) Pokud takový záznam neexistuje, paket je odeslán na výchozí bránu.
6. Příjemce přijme paket.
7. Příjemce aplikuje stejný postup od bodu 2 jako odesílatel.

Je důležité zmínit, že při každém průchodu přes směrovač je položka Time to live dekrementována o hodnotu 1. Pokud při nějakém průchodu je Time to live 0, paket je zahozen. Tento mechanizmus zabraňuje případnému zacyklení paketu v síti.

3.3.6 Způsob zobrazení protokolu v kurzu

V kurzu jsou popsány základní principy fungování tohoto protokolu. Základní vlastnosti jsou popsány textovou formou, principy směrování a fragmentace pro obě obě verze toho protokolu jsou znázorněny pomocí animace. Struktura IP paketu pro obě verze je znázorněna pomocí interaktivního datagramu, který vysvětluje, co která položka ve struktuře paketu znamená. V sekci cvičení jsou tyto animace konfigurovatelné a struktura datagramu rozšířená o další informace.

3.4 TCP protokol

Tato sekce obsahuje základní informace a principy protokolu TCP. Informace a principy jsou čerpány ze zdrojů [8, 9].
TCP protokol je spolehlivý, spojovaný a stavový protokol transportní vrstvy. TCP garantuje doručení všech segmentů ve správném pořadí, ale je náročnější na režii (TCP musí navázat, udržovat a následně ukončit spojení a také si vyžádat ztracené segmenty). Užití tohoto protokolu je vhodné všude, kde potřebujeme spolehlivý datový kanál. Jedná se například o elektronickou poštu, přenos souborů atd.

3.4.1 Struktura segmentu

Source port
: je port odesílatele

Destination port
: je port příjemce

Sequence number
je pořadové číslo prvního bajtu TCP segmentu v toku dat odesílatele k příjemci

Acknowledgment number
je číslo následujícího bajtu, který je příjemce připraven přijmout

Data offset
je délka záhlaví TCP segmentu

Reserved
je rezerva

Flags
je pole příznaků obsahující tyto příznaky:

URG
TCP segment nese naléhavá data

ACK
TCP segment má platnou položku Acknowledgment number (nastaven ve všech segmentech, kromě prvního segmentu, kterým klient navazuje spojení)

PSH
zpravidla se používá k signalizaci, že TCP segment nese aplikační data, příjemce má tato data předávat aplikaci, použití tohoto příznaku není ustáleno

RST
odmítnutí TCP spojení

SYN
odesílatel začíná s novou sekvencí číslování, TCP segment nese pořadové číslo prvního odesílaného bajtu

FIN
odesílatel ukončil odesílání dat

Window
je velikost okénka

Checksum
je kontrolní součet

Urgent pointer
je ukazatel naléhavých dat, k sequence number se přičte ukazatel naléhavých dat a výsledek ukazuje na konec úseku s naléhavými daty

Options
je pole obsahující volitelné položky záhlaví

Padding
je výplň

Data
položka obsahující přenášená data

3.4.2 Stavy[10]

CLOSED
defaultní stav, ve kterém jsou oba účastníci komunikace jestě před jejím navázáním

LISTEN
zařízení čeká na segment s příznakem SYN od druhého zařízení a jestě neodeslalo svůj segment s příznakem SYN

SYN_SENT
zařízení odeslalo segment s příznakem SYN a čeká na segment s příznakem SYN od druhého zařízení

SYN_RCVD
zařízení dostalo segment s příznakem SYN od druhého zařízení, odeslalo mu segment s příznakem SYN a čeká na jeho potvrzení

ESTABLISHED
spojení mezi zařízeními bylo úspěšně navázáno

CLOSE_WAIT
zařízení obdrželo segment s příznakem FIN a čeká na potvrzení požadavku na ukončení spojení od aplikace

LAST_ACK
zařízení již obdrželo a potvrdilo požadavek na ukončení spojení, odeslalo segment s příznakem FIN a čeká na jeho potvrzení

FIN_WAIT1
zařízení čeká na potvrzení svého segmentu s příznakem FIN nebo čeká na segment s příznakem FIN od druhého zařízení

FIN_WAIT2
zařízení má potvrzený jím odeslaný segment s příznakem FIN a čeká na segment s příznakem FIN od druhého zařízení

CLOSING
zařízení obdrželo segment s příznakem FIN a potvrdilo ho, odeslalo segment s příznakem FIN, který zatím nemá potvrzený

TIME_WAIT
zařízení obdrželo segment s příznakem FIN a potvrdilo ho, odeslalo segment s příznakem FIN a má ho potvrzený, zařízení čeká určitou dobu a poté je spojení ukončeno

3.4.3 Návazání spojení

Navázání spojení protokolu TCP užívá tvz. „three-way handshake“ princip. Navázání spojení protokolu TCP probíhá následujícím způsobem:

1. Iniciátor spojení je ve stavu CLOSED. Příjemce je ve stavu LISTEN.

2. Iniciátor zvolí náhodné sekvenční číslo a odešle segment s příznakem SYN. Iniciátor změní svůj stav na SYN_SENT.

3. Příjemce přijme tento segment, zvolí své sekvenční číslo a odešle segment s příznaky ACK a SYN. Acknowledgment number odesílaného segmentu je nastaveno na sekvenční číslo přijatého segmentu + 1. Příjemce změní svůj stav na SYN_RCVD

4. Iniciátor přijme tento segment a odešle segment s příznakem ACK. Sekvenční číslo odesílaného datagramu je nastaveno na acknowledgment number přijatého segmentu a acknowledgment number je nastaveno na sekvenční číslo přijatého segmentu + 1. Iniciátor změní svůj stav na ESTABLISHED.

5. Příjemce přijme tento segment a změní svůj stav na ESTABLISHED.

Tento postup je znázorněn na obrázku 3.2

3.4.4 Přenos

Pro samotný přenos se používá tzv. „sliding window“ technika. Odesílatel i příjemce již navázali spojení a jsou ve stavu ESTABLISHED.

1. Odesílatel dostane data od aplikační vrstvy.

2. Data jsou rozděleny na jednotlivé segmenty.

3. Odesílatel odešle stejný počet segmentů, jako je velikost okénka.

4. Příjemce přijímá segmenty potvrzuje je.

5. Pokud nějaký segment do určité doby nebyl potvrzen, je odeslán znovu.

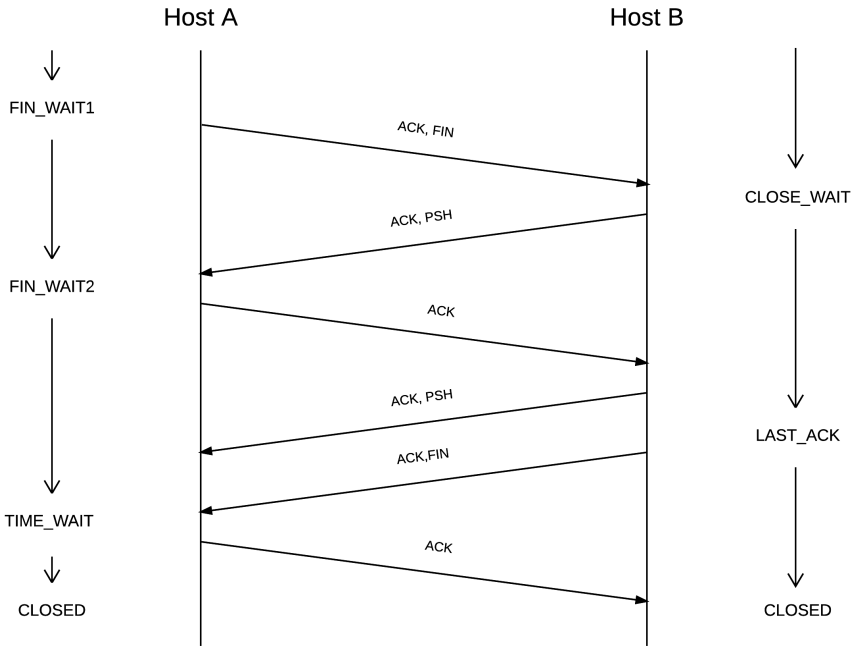
Obrázek 3.2: Navázání TCP spojení

3.4.5 Ukončení spojení

Ukončení spojení protokolu TCP používá tzv. „two-way handshake“. Tento princip je při ukončování spojení použit dvakrát. Ukončení spojení protokolu TCP probíhá následujícím způsobem:

1. Iniciátor ukončení spojení i příjemce jsou ve stavu ESTABLISHED
2. Iniciátor odešle stejný segment jako v průběhu spojení, akorát u něj nastaví příznak FIN. Iniciátor změní svůj stav na FIN_WAIT1.
3. Příjemce přijme tento segment a odešle segment s příznaky ACK a PSH. Příjemce změní svůj stav na CLOSE_WAIT.
4. Iniciátor přijme tento segment a změní svůj stav na FIN_WAIT2.
5. Dokončí se odesílání dat.
6. Příjemce odešle segment s příznaky ACK a FIN. Příjemce změní svůj stav na LAST_ACK.
7. Iniciátor přijme tento segment a odešle segment s příznakem ACK. Iniciátor změní svůj stav na TIME_WAIT a čeká určitou dobu.
8. Příjemce přijme tento segment a změní svůj stav na CLOSED.
9. Po uplynutí určité doby změní iniciátor svůj stav na CLOSED.

Tento postup je znázorněn na obrázku 3.3



Obrázek 3.3: Ukončení TCP spojení

3.4.6 Způsob zobrazení protokolu v kurzu

V kurzu jsou popsány základní principy fungování tohoto protokolu. Základní vlastnosti jsou popsány textovou formou, principy tzv. sliding window, navázání spojení a ukončení spojení jsou znázorněny pomocí animace. Struktura TCP segmentu je znázorněna pomocí interaktivního datagramu, který vysvětluje, co která položka ve struktuře paketu znamená. V sekci cvičení jsou tyto animace konfigurovatelné a struktura datagramu rozšířená o další informace.

3.5 UDP protokol

Tato sekce obsahuje základní informace a principy protokolu UDP. Informace a principy jsou čerpány ze zdrojů [11, 12].
UDP nespolehlivý, nespojovaný a bezstavový protokol transportní vrstvy.
UDP negarantuje doručení všech odeslaných datagramů a ani jejich doručení ve správném pořadí. Je méně náročný na režii a tudíž i rychlejší než protokol TCP. Užití tohoto protokolu je vhodné tam, kde je preferována rychlost přenosu a kde není třeba spolehlivý datový kanál. Jedná se například o streamovací služby, hry, DNS atd. Dalším vhodným využitím protokolu UDP jsou adresné oběžníky (multicast). Multicast neadresuje konkrétní cílový uzel, ale skupinu uzlů v síti (to protokolem TCP udělat nelze). Dochází tak k úspoře kapacity přenosových cest a ke snížení vytížení serveru (při použití TCP musí

každý účastník komunikace navázat a udržovat spojení se serverem, při užití UDP server šíří data pomocí těchto oběžníků). Šíření dat pomocí adresných oběžníků se používá například u multimediálních služeb.

3.5.1 Struktura datagramu

Source port
je port odesílatele

Destination port
je port příjemce

Length
vyjadřuje délku UDP datagramu

Checksum
je kontrolní součet

Data
položka obsahující přenášená data

3.5.2 Přenos dat

1. Odesílatel dostane data od aplikační vrstvy.
2. Data jsou rozděleny na jednotlivé datagramy.
3. Datagramy jsou jednotlivě odesílány příjemci.
4. Příjemce postupně přijímá jednotlivé datagramy.

3.5.3 Způsob zobrazení protokolu v kurzu

V kurzu jsou popsány základní principy fungování tohoto protokolu. Základní vlastnosti jsou popsány textovou formou, princip přenosu dat tohoto protokolu je znázorněn pomocí animace. Struktura TCP segmentu je znázorněna pomocí interaktivního datagramu, který vysvětluje, co která položka ve struktuře paketu znamená. V sekci cvičení jsou tyto animace konfigurovatelné a struktura datagramu rozšířená o další informace.

Kapitola 4

Návrh

4.1 GUI

Pro vytvoření grafického uživatelského rozhraní je v Javě na výběr ze tří grafických knihoven. Mezi tyto knihovny patří AWT, Swing a JavaFX. Pro návrh a následnou realizaci grafického uživatelského rozhraní kurzu byla vybrána knihovna JavaFX. Jako nástroj pro návrh a následnou realizaci byl použit nástroj JavaFX Scene Builder 2.0. Tento nástroj umožňuje rychlý a snadný návrh statických částí uživatelského rozhraní a následně generuje popis těchto částí ve formátu FXML, který lze použít v samotné aplikaci. Podrobnější popis knihovny JavaFX je v kapitole Realizace.

4.2 MVC

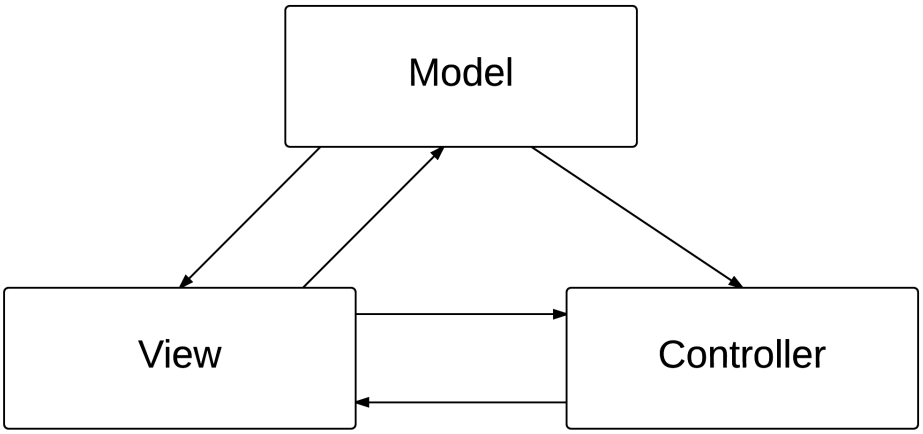
Pro vývoj aplikace byla vybrána architektura Model-view-controller. Tato architektura rozděluje aplikaci do tří nezávislých částí [13]. Architektura Model-view-controller je zobrazena na obrázku 4.1.

4.2.1 Model

Model má na starost správu doménové části aplikace. V případě této aplikace se model stará o načítání obsahu kurzu z XML souborů.

4.2.2 View

View definuje samotné uživatelské rozhraní. V případě této aplikace view obsahuje třídy, které jsou spjaty s generováním grafického rozhraní. Jedná se o třídy rozšiřující třídy z JavaFX a také třídy spjaté s načítáním grafického rozhraní z *FXML* souboru.



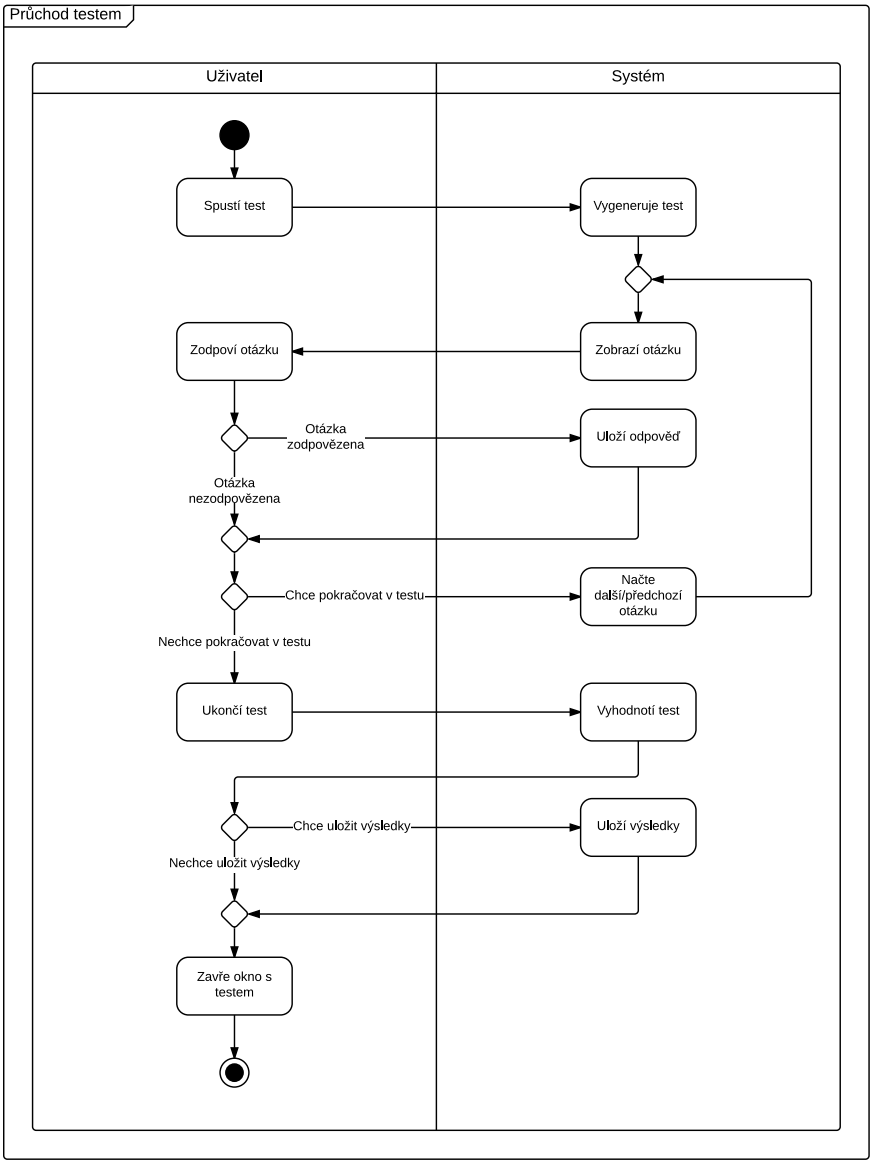
Obrázek 4.1: Model-view-controller

4.2.3 Controller

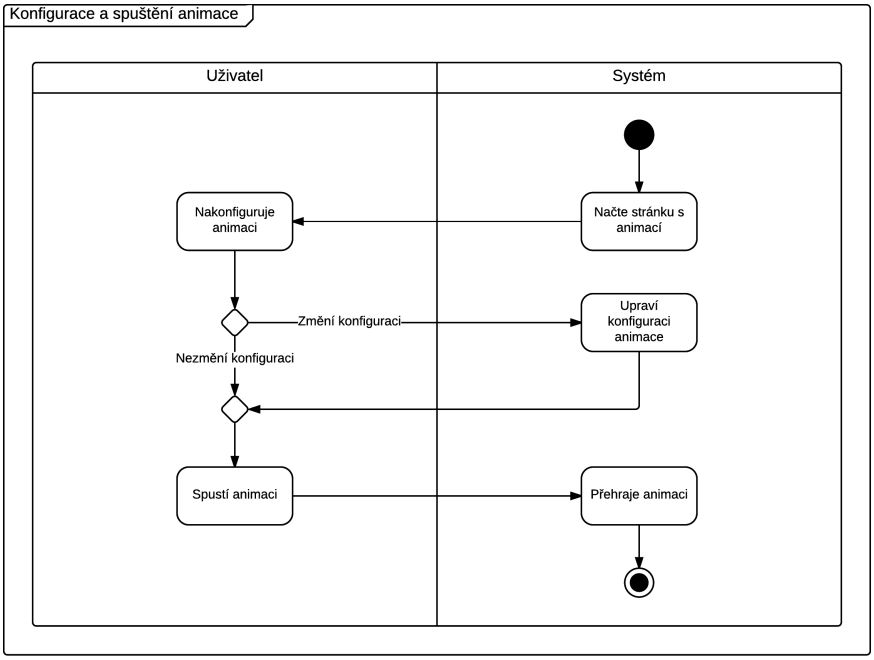
Controller má na starost obsluhu událostí vygenerovaných uživatelským rozhraním. V případě této aplikace má na starost obsluhu tlačítek a interaktivních prvků, dále má starost plnění a aktualizaci zobrazovaného obsahu.

4.3 Diagramy aktivit

Tato sekce je věnována diagramům aktivit. Diagram aktivit pro průběh testu je zobrazen na obrázku 4.2, diagram aktivit pro konfiguraci a spuštění animace na obrázku 4.3.



Obrázek 4.2: Diagram aktivit průběhu testu



Obrázek 4.3: Diagram aktivit konfigurace a spuštění animace

Kapitola 5

Realizace

5.1 Vývojové prostředí

Jako vývojové prostředí bylo zvoleno NetBeans. Hlavním důvodem této volby je, že s tímto vývojovým prostředím mám zatím největší zkušenost a také podporuje většinu použitých technologií.

5.2 Použité technologie

5.2.1 JavaFX

JavaFX je grafická knihovna Javy [14]. Jedná se o moderní nástroj pro vytváření grafického uživatelského rozhraní. Oproti knihovnám AWT nebo Swing přináší JavaFX velkou výhodu v možnosti popsání uživatelského rozhraní pomocí FXML souboru. Dále umožňuje popsání grafických komponent pomocí tříd stejným způsobem, jako v grafické knihovně Swing. V této aplikaci jsou k tvorbě uživatelského rozhraní využity oba tyto přístupy. Pro popis statických částí grafického rozhraní byl použit popis pomocí FXML souboru, pro dynamické části (např. konfigurovatelné animace) byly použity třídy, které na základě nějaké konfigurace generují příslušné grafické komponenty.

5.2.1.1 FXML

FXML je formát založen na formátu XML. Dále podporuje kaskádové styly. Umožňuje popsat grafické rozhraní zcela nezávisle na aplikační logice aplikace. Z pohledu architektury MVC představuje FXML soubor s popisem GUI část *View*. FXML umožňuje identifikovat jednotlivé komponenty grafického rozhraní pomocí atributu *fx:id*. Pomocí tohoto atributu lze v příslušném controlleru s těmito komponentami pracovat stejně, jako s proměnnými stejného typu. Proměnné odkazující na komponenty popsané v FXML souboru

jsou anotovány anotací *@FXML*. Příklad takového controlleru je uveden níže.

```
public class MyController implements Initializable {

    @FXML
    private Text myText;

    @Override
    public void initialize(URL location,
                           ResourceBundle resources) {
        myText.setText("Hello");
    }

}
```

5.2.2 Maven

Maven je nástroj pro správu projektů. Je založen na konceptu Project object model (POM). Tento model popisuje strukturu projektu (definuje základní adresářové struktury), definuje závislosti na externích knihovnách, popisuje buildovací proces aplikace atd. Sám Maven je modulární a jeho funkcionalita je dána pluginama, které se stahují z centrálních repozitářů Mavenu.

5.2.2.1 POM

Project object model projektu je popsán v souboru *pom.xml*. Tento soubor se nachází v kořenovém adresáři projektu a obsahuje minimálně koordináty projektu (v praxi obsahuje mnohem více informací např. závislosti, popis build procesu atd.) [15].

5.2.3 XML

Technologie XML je v tomto projektu použita zejména na popis obsahu kurzu. Jednotlivé textové stránky, animace, testové otázky a interaktivní struktury datagramů jsou popsány v příslušných resource souborech. Úpravou příslušného XML resource souboru tak lze snadno přidat, odebrat nebo upravit konkrétní stránku v kurzu nebo cvičení. V případě testových otázek lze úpravou konkrétního XML souboru přidat, upravit nebo smazat otázku z poolu, ze kterého se samotné testy generují. Příklad použití technologie XML pro popis obsahu kurzu je popsán níže.

```
<content>
    <page>
        <title>Title</title>
        <line>First line</line>
        <line>Second line</line>
    </page>
</content>
```

5.2.3.1 SAX

SAX (Simple API for XML) je technologie pro parsování XML souborů [16], která je v tomto projektu použita na parsování resource souborů s obsahem kurzu. Samotné parsování XML dokumentu provádí metoda *parse* třídy *SAXParser*. Této metodě je v parametrech předán XML soubor a handler, který definuje akce, které se při parsování provadí. Handler rozšiřuje třídu *DefaultHandler* a obsahuje tyto tři přepsané metody:

startElement
: metoda, definující akci při nalezení počátečního tagu

endElement
: metoda, definující akci při nalezení koncového tagu

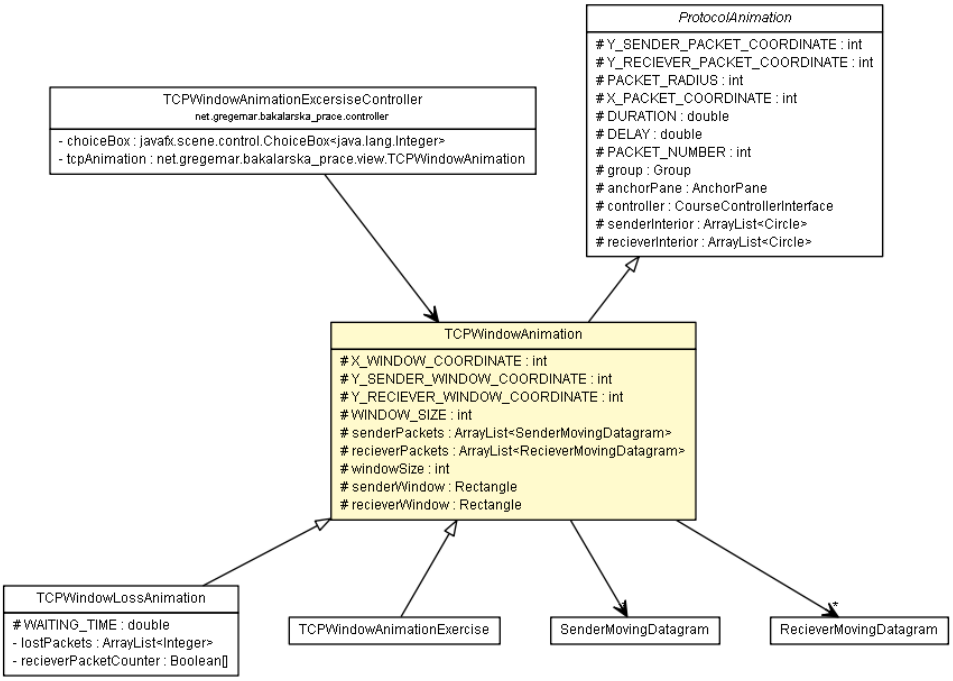
characters
: metoda, definující akci při nalezení obsahu mezi tagy

Příklad handleru je zobrazen níže.

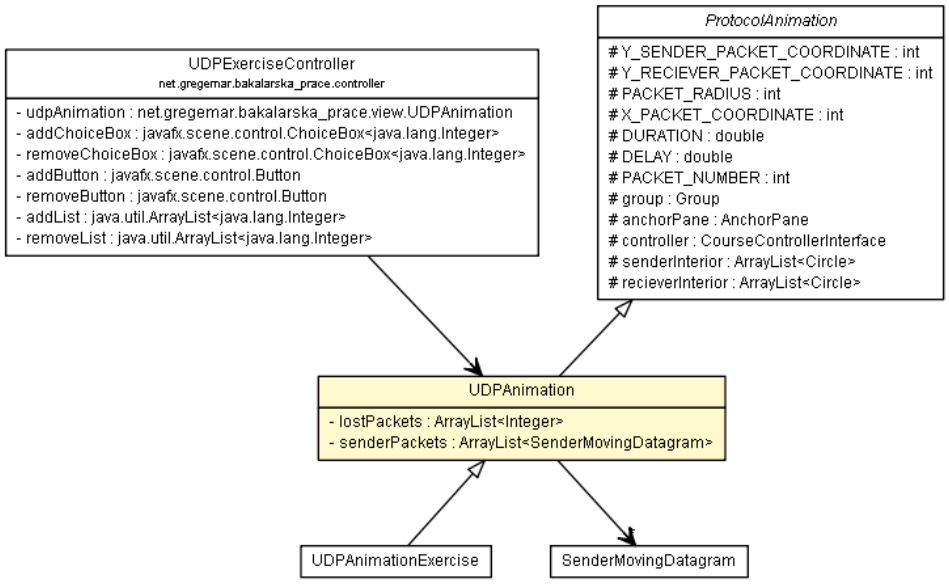
```
public class XMLHandler extends DefaultHandler {

    @Override
    public void startElement(String uri,
                             String localName,
                             String qName,
                             Attributes attributes)
                             throws SAXException {
        //start element action
    }

    @Override
    public void endElement(String uri,
                           String localName,
                           String qName)
                           throws SAXException {
        //end element action
    }


    @Override
    public void characters(char ch[],
                           int start,
                           int length)
                           throws SAXException {
        //text action
    }

}
```

5.3 Animace přenosu dat protokolů TCP a UDP

Animace chování protokolu TCP/UDP je v kurzu reprezentována samostatnou třídou. Tato třída má na starost veškerou činnost potřebnou pro vytvoření, spuštění, konfiguraci a zastavení konkrétní animace. Tato třída je také plně konfigurovatelná pomocí příslušného xml souboru nebo v sekci cvičení pomocí choiceBoxů, které má uživatel k dispozici. Pro tvorbu samotných animací jsou v kurzu použity podtřídy třídy *Transition*. Diagram tříd pro animace TCP protokolu je zobrazen na obrázku 5.1, pro animace UDP protokolu na obrázku 5.2.



Obrázek 5.1: Diagram tříd pro TCP animace



Obrázek 5.2: Diagram tříd pro UDP animace

5.4 Interaktivní struktura datagramu

Popis struktury datagramu konkrétního protokolu je dána FXML souborem popisující samotný vzhled této struktury a XML souborem, který obsahuje položky této struktury obsahující informace o dané položce. Controller, který obsahuje informace o všech položkách datagramu, pak na každé zobrazené položce struktury datagramu odchytává následující události:

OnMouseEntered
: Tato událost definuje akci při najetí myší na položku. V našem případě controller zobrazí informace o položce.

OnMouseExited
: Tato událost definuje akci při najetí myši pryč z položky. V našem případě controller schová informace o položce.

5.5 Testové rozhraní kurzu

Otázky a odpovědi s ohodnocením pravdivosti jsou načítány s příslušných XML resource souborů. Tímto způsobem lze v případě nutnosti celkem pohodlně přidávat, mazat nebo upravovat tyto otázky bez nutnosti měnit zdrojový kód systému. Každá otázka ve svém popisu obsahuje tag *answerType*. Na základě tohoto tagu pak systém generuje typ odpovědi na danou otázku. Těmito typy odpovědí jsou myšleny:

- odpověď pomocí radioButtonu
- odpověď pomocí checkBoxu
- odpověď pomocí textFieldu

Vygenerovaný test pak obsahuje sadu náhodně vybraných otázek.

5.6 Testování

Testování aplikace lze rozdělit na dvě části:

- testování aplikační logiky
- testování grafických komponent

5.6.1 Testování aplikační logiky

Testy aplikační loginy jsou realizovány pomocí frameworku JUnit. Tento framework má ve vývojovém prostředí NetBeans podporu, tudíž je tvorba těchto testů jednoduchá.

5.6.1.1 Konfigurace testu

Pro vybranou třídu je vytvořena třída implementující testy. Tato třída obsahuje samotné testy vybraných metod. Tyto metody jsou anotovány pomocí anotace *@Test*. Na ověření správného chování testovaných metod jsou použity metody třídy *org.junit.Assert*.

5.6.2 Testování grafických komponent

Testy grafických komponent jsou realizovány pomocí frameworku TestFX. Tento framework není ve výjovovém prostředí NetBeans přímo podporován, avšak přidání tohoto frameworku do projektu není složité, protože je umístěn v centrálním repozitáři Mavenu. Pro přidání tohoto frameworku je nutné upravit *pom.xml* soubor projektu přidáním závislosti projektu na tomto frameworku. Pro přidání závislosti se používá atribut *dependency*, ve kterém jsou umístěny koordináty tohoto frameworku. Pro tento projekt byla vybrána verze 3.1.0 tohoto frameworku. Je to dáno hlavně tím, že verze 4 je stále v alfa verzi.

5.6.2.1 Konfigurace testu

Pro vybranou třídu je vytvořena třída implementující testy. Tato třída rožšiřuje třídu *GuiTest*. Třída s testy musí implementovat metodu *getRootNode*, která načítá samostatné grafické komponenty k otestovaní. Dále třída *GuiTest* poskytuje třídě s testy metody na simulaci chování uživatele v GUI (např. kliknutí na tlačítko, přesunutí kurzoru na pozici atd.). Dále umožňuje identifikovat jednotlivé podkomponenty v testované komponentě (např. pomocí id). Samotná metoda provádějící test je anotována anotací *@Test*. Pro ověření správnosti chování testované grafické komponenty lze použít metody poskytnuté tímto frameworkem (např. metoda *verifyThat*), nebo metody třídy *org.junit.Assert*.

```
public class GuiTestClass extends GuiTest{

    @Override
    protected Parent getRootNode() {
        // create tested node
    }

    @Test
    public void guiTest() {
        // test created node
    }

}
```

5.7 Dokumentace

Pro vytvoření dokumentace byl použit nástroj JavaDoc. Tento nástroj je integrován do vývojového prostředí NetBeans.

Závěr

Osobní přínos

Osobní přínos této práce byl obrovský. Měl jsem možnost se detailněji seznámit s použitými technologiemi. Jedná se převážně o bližší pochopení fungování grafické knihovny JavaFX a zjištění, že testování této knihovny není úplně triviální. Dále jsem se během vývoje tohoto systému naučil věci, které jsou při vývoji použitelné a naopak věci, které vývoj zpomalovaly a znesnadňovaly a je jim se dobré v příštích projektech vyvarovat.

Zhodnocení cílů

Byl vytvořen interaktivní vzdělávací kurz, který splňuje v zadání a námi definované požadavky. Kurz vysvětluje principy jednotlivých protokolů názorným způsobem pomocí interaktivních prvků. Dále je dbáno na snadnou rozšiřitelnost kurzu. To je dosaženo použitím XML pro popis obsahu kurzu.

Výhled do budoucna

V nejbližší době bude tento vzdělávací kurz zveřejněn na portálu rvp.cz. Pokud ohlasy na tento kurz budou kladné, je možné rozšířit tento kurz o další protokoly, které v kurzu zatím chybí. V případě nejasností při vysvětlování principů fungování protokolů je možné kurz doplnit o další prvky vysvětlující danou problematiku.

Literatura

[1] Moodle: Co je Moodle [online]. Prosinec 2006, [cit. 2015-04-17]. Dostupné z: https://docs.moodle.org/archive/cs/Co_je_Moodle

[2] Kozierok, C. M.: TCP/IP Architecture and the TCP/IP Model [online]. Září 2005, [cit. 2015-04-12]. Dostupné z: http://www.tcpipguide.com/free/t_TCPIPArchitectureandtheTCPIPModel-2.htm

[3] Braden, R.: Requirements for Internet Hosts - Communication Layers. Technická zpráva, RFC Editor, Říjen 1989, [cit. 2015-04-17]. Dostupné z: http://www.rfc-editor.org/rfc/rfc1122.txt

[4] 5 IP protokol [online]. [cit. 2015-04-17]. Dostupné z: http://zam.opf.slu.cz/botlik/CD-0x/5.html

[5] Postel, J.: Internet Protocol. Technická zpráva, RFC Editor, Září 1981, [cit. 2015-04-25]. Dostupné z: http://www.rfc-editor.org/rfc/rfc791.txt

[6] IP nové generace [online]. [cit. 2015-04-17]. Dostupné z: http://zam.opf.slu.cz/botlik/CD-0x/8.html

[7] Deering, S. E.; Hinden, R. M.: Internet Protocol, Version 6 (IPv6) Specification. Technická zpráva, RFC Editor, Prosinec 1998, [cit. 2015-04-25]. Dostupné z: http://www.rfc-editor.org/rfc/rfc2460.txt

[8] 9 Protokol TCP (Transmision Control Protocol) [online]. [cit. 2015-04-17]. Dostupné z: http://zam.opf.slu.cz/botlik/CD-0x/9.html

[9] Postel, J.: Transmission Control Protocol. Technická zpráva, RFC Editor, Září 1981, [cit. 2015-04-25]. Dostupné z: http://www.rfc-editor.org/rfc/rfc793.txt

[10] Kozierok, C. M.: TCP Operational Overview and the TCP Finite State Machine (FSM) [online]. Září 2005, [cit. 2015-04-12]. Dostupné z: http://www.tcpipguide.com/free/t_TCPOperationalOverviewandtheTCPFiniteStateMachineF-2.htm

[11] 10 Protokol UDP (User Datagram Protocol) [online]. [cit. 2015-04-17]. Dostupné z: http://zam.opf.slu.cz/botlik/CD-1x/10.html

[12] Postel, J.: User Datagram Protocol. Technická zpráva, RFC Editor, August 1980, [cit. 2015-04-25]. Dostupné z: http://www.rfc-editor.org/rfc/rfc768.txt

[13] Ootips: Model-View-Controller [online]. Květen 1998, [cit. 2015-04-25]. Dostupné z: http://ootips.org/mvc-pattern.html

[14] Oracle: *Java Platform, Standard Edition (Java SE) 8 [online]*. [cit. 2015-04-20]. Dostupné z: http://docs.oracle.com/javase/8/javase-clienttechnologies.htm

[15] The Apache Software Foundation: *Introduction to the POM [online]*. [cit. 2015-04-22]. Dostupné z: https://maven.apache.org/guides/introduction/introduction-to-the-pom.html

[16] About SAX [online]. [cit. 2015-04-12]. Dostupné z: http://www.saxproject.org/about.html

Příloha **A**

Seznam použitých zkratek

GUI Graphical user interface

ICMP Internet Control Message Protocol

IP Internet protocol

MTU Maximum transmission unit

MVC Model-view-controller

POM Project object model

SAX Simple API for XML

TCP Transmission control protocol

UDP User datagram protocol

XML Extensible markup language

Příloha B

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou formou implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF
- `doc`
  - `manual.pdf` ..... uživatelská dokumentace ve formátu PDF
  - `app` ..... adresář s dokumentací zdrojových kódů kurzu
  - `test` ..... adresář s dokumentací zdrojových kódů testů