Implementace klienta NNTP

Ing. Filip Merhaut

Bakalářská práce
2006

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav automatizace a řídicí techniky
akademický rok: 2005/2006

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Ing. Filip MERHAUT**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Automatické řízení a informatika**
Téma práce:	**Implementace klienta NNTP**

Zásady pro vypracování:

Navrhněte a realizujte klienta NNTP protokolu. K implementaci využijte platformy .NET a vývojové prostředí Visual Studio .NET. Práce bude také obsahovat stručný přehled historie a specifikací NNTP protokolu.

Klient bude realizován metodou objektově orientovaného programování. Ke své činnosti bude používat technologii více vláken – multithreading. Uživatelské rozhraní aplikace bude mít formu GUI – Graphical User Interface.
Klient bude obsahovat následující funkce:
- Připojení a autentizace na server
- Stažení seznamu skupin dostupných na serveru
- Stažení seznamu příspěvků ve skupině
- Čtení/zasílání příspěvků
- Možnost ukládat binární přílohy

Rozsah práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

- **Blum Richard: Network Programming, Sybex 2003, ISBN: 0782141765**
- **Kabelová Alena, Dostálek Libor: Velký průvodce protokoly TCP/IP a systémem DNS, 3. aktualizované vydání, Computer Press, a. s. 2002, ISBN: 80-7226-675-6**
- **Titus Tobin: Threading Handbook, APress LLC 2004, ISBN: 1861008295**
- **RFC 850 Standard for Interchange of USENET messages [online] dostupnost z www: http://rfc.net/rfc850.html**
- **RFC 977 Network News Transfer Protocol [online] Dostupnost z www: http://rfc.net/rfc977.html**

Vedoucí bakalářské práce: **Ing. Miroslav Červenka**
Ústav aplikované informatiky

Datum zadání bakalářské práce: **14. února 2006**

Termín odevzdání bakalářské práce: **26. května 2006**

Ve Zlíně dne 14. února 2006

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
L.S.

prof. Ing. Vladimír Vašek, CSc.
pověřený děkan

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

ABSTRAKT

Tato bakalářská práce se zabývá vývojem softwarového produktu. Cílem bylo vytvořit plně funkčního uživatelského klienta pro práci s protokolem NNTP pod operačním systémem MS Windows. NNTP je síťový protokol textového typu. Je základem sítě Usenet, která je jednou z nejstarších částí Internetu. Jejím účelem je poskytovat prostředky pro komunikaci v uživateli vytvářených diskusních skupinách, zaměřených převážně na odborná témata. Tyto diskusní skupiny jsou jakousi globální obdobou emailových konferencí. Program byl vytvořen v jazyce C#, s využitím principů objektového programování a technologií .NET Framework.

Klíčová slova: počítačové sítě, objektové programování, NNTP, Usenet, .NET, C#

ABSTRACT

This bachelor thesis deals with the development of a software product. The objective was to create a fully functional user client that works with the NNTP protocol under the MS Windows operating system. The NNTP is a text-based protocol. It is the base on which the Usenet network, one of the oldest parts of the Internet, stands. Its purpose is to provide means that enable the clients to communicate in the user-created discuss groups, focused mainly on the technical subjects. This newsgroups system is the analogy of the mailing-lists. The program was developed in the C# language, with utilization of the object-oriented programming and the .NET Framework technology.

Keywords:computer networks, object-oriented programming, NNTP, Usenet, .NET, C#

Děkuji tímto vedoucímu mé bakalářské práce Ing. Miroslavu Červenkovi, za odborné vedení, rady, připomínky a čas, který mi věnoval při zpracování této bakalářské práce.

Souhlasím s tím, že s výsledky mé práce může být naloženo podle uvážení vedoucího bakalářské práce a ředitele ústavu. V případě publikace budu uveden jako spoluautor.

Prohlašuji, že jsem na celé bakalářské práci pracoval samostatně a použitou literaturu jsem citoval.

Ve Zlíně …….

Podpis diplomanta

OBSAH

ÚVOD

Tato práce se zabývá tvorbou softwarového produktu umožňujícího práci se sítí Usenet. Usenet je jedna z rozsáhlých podsítí dnešního Intenetu. Pracuje na principu klient-server. Je plně decentralizovaná ve smyslu neexistence žádného fyzického řídícího uzlu ani centrální autority. Rozsah Usenetu je globální. Jeho úlohou je poskytovat uživatelům rozhraní pro čtení a zasílání zpráv v rámci hierarchicky členěných diskusních skupin. Zprávy jsou po určitou dobu uchovávány na serverech odkud si je uživatelé stahují k sobě do počítače. Zpráva zaslaná uživatelem na místní server se v řádech minut až hodin rozšíří na všechny Usenetové servery v síti a je tím pádem dostupná pro ostatní uživatele kdekoliv na světě. Usenet pracuje na základě protokolu NNTP – Network News Transfer Protocol. Hlavním úkolem této bakalářské práce bylo implementovat klienta NNTP protokolu.

K realizaci byla zvolena vývojová technologie .NET Framework, konkrétně jazyk C#. K výhodám technologií .NET patří rozsáhlá množina základních tříd, automatická správa paměti, snadno dostupná online podpora pro vývojáře, možnost snadno implementovatelného grafického rozhraní v OS Windows a nativní podpora moderních technologií jako jsou Unicode a XML. Na druhou stranu jsou tyto výhody vykoupeny o něco nižším výkonem výsledné aplikace, neboť .NET programy nejsou kompilovány do strojového kódu konkrétního procesoru, nýbrž do mezikódu MSIL. Tento mezikód je do strojového kódu procesoru kompilován až při takzvané Just In Time kompilaci, která proběhne při spuštění programu. Jedná se tedy o podobný proces jako například interpretace jazyka Java v Java Virtual Machine. I přes značné úsilí firmy Microsoft o optimalizaci tohoto procesu jsou programy vyvíjené například pomocí C++ a WinAPI stále výrazně rychlejší. V práci je rovněž nastíněna historie a současnost sítě Usenet a podrobný rozbor protokolu NNTP.

I. TEORETICKÁ ČÁST

1 USENET

Usenet je celosvětová decentralizovaná síť typu klient/server. Je to síť, jenž je svým charakterem velmi blízká příbuzná emailových konferencím. Je zde však jeden podstatný rozdíl. Zatímco v emailové konferenci je každý zaslaný email zaslán do schránky všem účastníkům konference, v Usenetu je každá zaslaná zpráva rozeslána pouze na všechny servery, kde je po určitou dobu (obvykle jeden měsíc) archivována. Teprve ze svého místního serveru si ji pak jednotliví účastníci diskuse vyzvednou. Výhoda tohoto přístupu je v tom, že je úspornější z hlediska přenosového pásma a z hlediska zahlcení schránek klientů zprávami, které je nezajímají. Klienti Usenetu jsou tedy uživatelé, kteří v něm prostřednictvím serverů vyhledávají a čtou zprávy od ostatních uživatelů a také do něj zprávy pro ostatní uživatele zasílají. Úkolem každého serveru je přijímat tyto zprávy, zařadit je do příslušné diskusní kategorie, po určitou časovou dobu je uchovávat, zasílat je ostatním serverů usenetové síti tak aby se zpráva bezodkladně rozšířila po celém světě a hlavně poskytovat svým klientům k přístup k těmto zprávám. Protokol na základě kterého veškerá tato komunikace probíhá se nazývá NNTP – Network News Transfer Protocol. Jedná se o textový protokol schopný pracovat s proudy dat. [2]

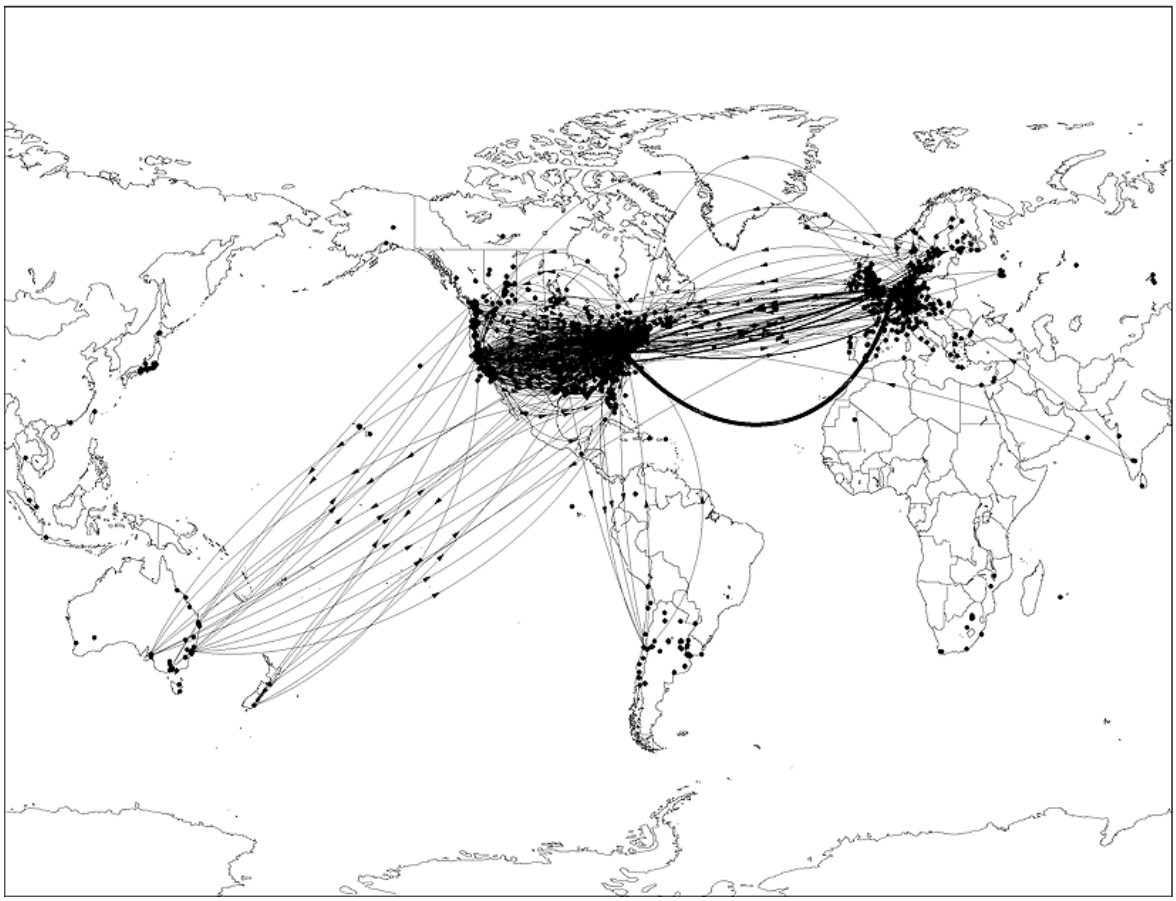
1.1 Historie Usenetu

Vznik Usenetu se datuje do roku 1979, krátce po vydání operačního systému Unix V7, který jako jednu z novinek obsahoval síťový protokol UUCP. Dva postgraduální studenti z Duke Univerzity v Severní Karolíně, Tom Truscott a Jim Ellis, přišli s nápadem propojit univerzitní počítače tímto protokolem tak, aby mezi nimi bylo možné vyměňovat zprávy a soubory podobně tak jako v emailových konferencích v tehdy ještě vojenském ARPANETu. Jejich nápadu se chopil další postgraduální student, Steve Bellovin, který vytvořil úplně první Netnews program ve skriptovacím jazyce Bourne shell. Síť, kterou tito studenti vytvořili, měla v počátku pouhé dva počítače a první hierarchií, která v ranném Usenetu vznikla byla net.*, kde se diskutovalo o síti samotné. Popularita tohoto projektu se v akademickém prostředí velmi rychle šířila. Vznikl nový veřejný prostor, který brzy dosáhl kritické masy dost vlivné na to, aby se do něj během dvou let zapojily univerzity z celého severoamerického kontinentu. [6]

Další milník přišel v roce 1984. Na Kalifornské Univerzitě v San Diegu vzniknul protokol NNTP. Od této chvíle se Usenetová komunikace začala šířit v mnohem efektiv-

nějším a spolehlivějším protokolu TCP/IP. Tento protokol také zaručil přežití Usenetu jako součásti dnešní podoby Internetu. NNTP protokol byl později standardizován v dokumentu RFC 977. Na přelomu let 1986/1987 došlo k události nazývané „Velké Přejmenování“, která dala vznik dnešní podobě názvů v hierarchickém systému diskusních skupin. Další milník přišel roce 1990, ve kterém byl publikován dokument RFC 2980. Ten významně rozšiřuje původní skupinu NNTP příkazů. Od konce osmdesátých do první poloviny devadesátých let začalo postupné pronikání Usenetu i mimo akademickou sféru, ostatně jako celého Internetu. Příznačné pro tento jev bylo spuštění služeb firmy AOL - America OnLine, největšího internetového providera v USA 90. let. Tímto rozšířením mezi masy také bohužel došlo k rozmělnění dřívější pospolité atmosféry v Usenetu a ke všeobecnému úpadku síťové etiky. Davy nových nezkušených uživatelů nerespektovaly žádná pravidla. Na dobu před jejich příchodem často účastníci původního Usenetu vzpomínají jako na „zlatý věk“. Od poloviny devadesátých let je charakteristická tendence nárůstu počtu binárních skupin, které neslouží ani tak k diskutování o konkrétních tématech jako spíše k šíření binárních souborů. Přelom tisíciletí přinesl velký problém v podobě spamu, který od té doby v masivní míře otravuje komunikaci téměř ve všech diskusních skupinách. Zatím poslední velkou událostí týkající se Usenetu je spuštění služby GoogleNews firmy Google na počátku roku 2001. Je to ambiciózní projekt, který poskytuje webové rozhraní k Usenetu pro ty, kteří nemají přístup k žádnému ze serverů. Jeho hlavním přínosem ale je snaha poskytnout přístup ke kompletnímu archivu Usenetové komunikace od roku 1981 (tedy pouze textových zpráv, binární přílohy přirozeně zaznamenávány nejsou). S trochou snahy tak lze nalézt desítky let staré původní zprávy, které v mládí do Usenetu zaslali mnohé v současnosti slavné a významné osobnosti počítačového průmyslu.

Rozsah této sítě lze v dnešní době jen odhadnout. Jediné co můžeme konstatovat je, že objem přenesených dat, měřený v bytech za den, se každý rok přibližně zdvojnásobuje. V dnešní době se pohybuje v řádech jednotek terabytů za den.



Obr. 1. Mapa hlavních Usenetových uzlů v roce 1993

V ranných dobách se ještě vytvářely mapy sítě - orientované grafy, ve kterých uzly představují jednotlivé servery a hrany mezi nimi přenosové spoje. Při dnešním počtu serverů by takovou mapu nebylo prakticky proveditelné sestavit. Její poslední známá verze je více než deset let stará. [11]

1.2 Klientský software

NNTP je běžný znakový protokol, podobný například protokolu HTTP. K práci s ním tedy v podstatě stačí běžný TELNETový terminál, který je k dispozici v každém rozumném operačním systému. Například ve všech verzích MS Windows jej lze spustit příkazem telnet.exe z příkazové řádky. Takový způsob práce sice možný je, pro praktické použití je velmi nepohodlný. V této práci je nicméně využíván pro názornost a možnost tak nahlédnout „pod pokličku“ toho co se děje v programech s grafickým rozhraním.

K pohodlnému procházení, čtení, stahování, ukládání a zasílání usenetových příspěvků a jejich příloh je vytvořena celá řada softwarových klientů pro všechny myslitelné

platformy. Některé fungují v textovém módu, většina nabízí grafické uživatelské rozhraní. O tvorbě jednoho takového klienta s grafickým rozhraním pojednává tato práce.

1.3 Serverový software

NNTP server je poměrně komplikovaný software. Musí být schopen zvládat masivní síťový provoz, vyřizování mnoha požadavků najednou a rychlou práci s velkým množstvím souborů. Dominantním řešením dnešních Usenetových serverů je program INN (InterNetNews), běžící na operačním systému Linux. Dále je oblíbený také Zilla-NNTP. Z hlediska platformy Windows je nejrozšířenější NNTP service, běžící jako součást serverových distribucí OS Windows. Kromě těchto nejznámějších řešení existují desítky malých serverů, často volně šiřitelných. K testování aplikace vytvořené v rámci této práce byl použit server Dnews.

2 NNTP

NNTP je zkratka pro Network News Transfer Protocol. Servery poskytující služby založené na tomto protokolu naslouchají na portu 119. Správce konkrétního serveru přirozeně může nastavit naslouchání na libovolném volném portu, nicméně obecná praxe je právě port 119. NNTP je výhradně textový protokol. Příkazy a odpovědi se tedy skládají ze 128 znaků ASCII tabulky. Prvních 33 znaků je „netisknutelných“ a v počítačových systémech mají většinou význam příkazových znaků, zbylých 95 znaků je tisknutelných. K přenosu tohoto textového protokolu je tedy potřeba pouhých 7 bitů. Je-li přenosový kanál protokolu 8 bitový (což většinou je), je každý sedmibitový znak zarovnán doprava a nejvyšší bit oktetu nastaven na nulu. Díky znakovému charakteru protokolu nicméně vznikají potíže při přenosu netextových dat – binárních souborů v rámci přílohy zprávy. Podrobněji se tomuto problému a jeho řešení věnuji v části věnované kódování binárních příloh.

2.1 Formální definice protokolu

Protokol NNTP je definován v příslušných dokumentech RFC. RFC je jedním ze základních pilířů dnešního Internetu a vzájemné kompatibility jeho součástí. RFC je zkratka anglického Request for comments (žádost o komentáře). Tato zkratka se používá k označení řady standardů a dalších dokumentů popisujících internetové protokoly a systémy. RFC jsou formálně označeny jako doporučení, přesto se podle nich řídí drtivá většina internetových systémů. Neexistuje žádná formální moc která by jejich dodržování vymáhala. Pokud však některý internetový systém příslušené RFC normy ze své oblasti nedodržuje, riskuje nekompatibilitu se zbytkem souvisejících systémů na internetu.

Všechna RFC lze volně získat na adrese http://www.ietf.org/rfc.html. Každé RFC je dostupné v podobě čistého anglického ASCII textu. Původními autory jednotlivých RFC jsou obvykle experti v daném oboru, kteří se snaží řešit konkrétní problém. Jeho řešení pak nabídnout internetové komunitě v podobně návrhu RFC. Pokud je dané řešení přínosné, dokument se vydá jako RFC. První RFC dokument byl vydán 7. dubna 1967. V době psaní této práce (květen 2006) se pořadové číslo nejnovějšího RFC dokumentu (ne)zastavilo na hodnotě 4501.

Původní sada příkazů NNTP je popsána v dokumentu RFC 977 (únor 1986). Kořeny dnešního Usenetu však lze nalézt již v dokumentu RFC 850 (červen 1983), který popisuje výměnu „news“ zpráv v tehdejším ARPANETu. Standard specifikovaný v RFC 850 je

dostačující k základní komunikaci typu klient-server a server-server. V mnoha ohledech ale jeho funkcionalita po čase nestačila nebo nebyla efektivní. S tím jak se Usenet v 80. letech stával více a více populární, rostla také potřeba tyto nedostatky odstranit. V roce 1991 začaly práce na formální revizi standardu, ale nebyly nikdy dokončeny. Navzdory tomu bylo v následujících letech mnoho konceptů z této práce neformálně adaptováno tvůrci nejrůznějšího Usenetového software. Tito tvůrci také často přidávali do svých programů nestandardní rozšíření protokolu. Některé z těchto rozšíření se díky svému masivnímu rozšíření staly standardem de facto. [7], [8]

Koncem 90. let proto většina Usenetového software implementovala různé varianty NNTP, jejichž možnosti dalece přesahovaly původní návrh protokolu. Přirozeně ne každý usenetový program podporoval úplně všechny „extra“ funkce, což vedlo k potenciální nekompatibilitě mezi klienty a servery. V říjnu 2000 byla tedy konečně vydán dokument RFC 2980, Common NNTP Extensions (Nejčastější rozšíření NNTP), který tyto vlastnosti formalizoval v souladu s původní RFC 977. [10]

NNTP rozšíření se skládají hlavně z několika nových NNTP příkazů které jsou přidány do základní sady a z menších úprav funkcionality některých původních příkazů. Rozšíření spadají do některé z těchto tří kategorií: ty které zvyšují efektivitu přenosu zpráv mezi servery, ty které zvyšují efektivitu přístupu klientů ke zprávám a zbylé, které nespadají do žádné z těchto dvou kategorií.

Dokumenty RFC 977 a RFC 2980 jsou základní normy, ze kterých bylo vycházeno při tvorbě této práce.

2.2 Příkazy protokolu

Příkaz se skládá z klíčového slova a případně dalších parametrů. Každý parametr je oddělen mezerou nebo jiným prázdným znakem. Příkaz se ukončí odřádkováním – znak CR-LF (carriage return, line feed, někdy také označován jako ‘\r\n’). Příkazy nejsou citlivé na velikost písma. Maximální délka příkazu na jednom řádku je 512 bytů včetně odřádkování.

2.2.1 Příkazy definované v RFC 977

RFC 977 specifikuje základní příkazy které musí povinně implementovat každý server v Usenetu. Jsou uvedeny v následující tabulce. [8]

Tab. 1. NNTP příkazy dle RFC 977

ARTICLE [<message-id> | number]
BODY [<message-id> | number]
GROUP newsgroup
HEAD [<message-id> | number]
HELP
IHAVE
LAST
LIST
NEWGROUPS yyyymmdd hhmmss
NEWNEWS newsgroups yyyymmdd hhmmss
NEXT
POST
QUIT
SLAVE
STAT [<message-id> | number]

2.2.2 Příkazy definované v RFC 2980

RFC 2980 rozšiřuje množinu NNTP příkazů. Nejsou to však z hlediska NNTP protokolu příkazy povinné, pouze poskytují funkcionalitu navíc. Málokterý server je implementuje všechny. Všechny tyto příkazy vznikly „samovolně“ jako nápady tvůrců jednotlivých NNTP klientů/serverů a byly kodifikovány dodatečně na základě toho že se dočkaly širokého rozšíření. Jsou uvedeny v následující tabulce. [10]

Tab. 2. NNTP příkazy dle RFC 2980

CHECK <message-id>
MODE [STREAM | READER]
TAKETHIS <message-id>
XREPLIC ggg:nnn[,ggg:nnn…]

LIST [ACTIVE | ACTIVE.TIMES | DISTRIBUTIONS | DISTRIBUTIONS.PATS | NEWSGROUPS | LIST OVERVIEW.FMT | SUBSCRIPTIONS]
LISTGROUP
XGTITLE
XHDR field [range]
XINDEX
XOVER
XPAT
XPATH <message-id>
XOVER [range]
XTHREAD
AUTHINFO [USER | PASS | SIMPLE | GENERIC]
DATE
WILDMAT

2.2.3 Podrobný popis jednotlivých příkazů

ARTICLE – požadavek na zaslání celého textu konkrétní zprávy. Existují dvě formy příkazu, lišící se tím jak chceme konkrétní požadovaný příspěvek specifikovat. Buď podle jeho ID nebo podle jeho pořadového čísla ve skupině. Rozdíl mezi nimi je následující: ID je unikátní identifikátor, který je jedinečný pro každou zprávu v Usenetu. Je proto velmi dlouhý (a také pro člověka špatně čitelný). ID je vytvořeno při zasílání zprávy a je její neměnnou částí po celou dobu její existence. Zatímco pořadové číslo zprávy ve skupině je identifikátor který je platný pouze na daném serveru. Jedna a tatáž zpráva má na dvou různých serverech dvě naprosto odlišná čísla. Pořadové číslo může být velmi malé, u serverů kde je malá populace zpráv ve skupině se může jednat třeba jen o dvoumístné číslo. Na druhou stranu existují servery jež poskytují skupiny s miliony zpráv, v takovém případě bývá pořadové číslo zprávy třeba i osmimístné (každý server začíná při svém uvedení do provozu číslo-

vat zprávy od jedničky, nicméně toto počítadlo obvykle nebývá resetováno, proto čísla zpráv rostou na stamiliony i když skupina obsahuje pouze setiny tohoto počtu zpráv).

BODY – chová se stejně jako příkaz ARTICLE. Jediný rozdíl je v tom že nevrátí celý text zprávy ale pouze její tělo bez hlaviček.

CHECK - při připojení typu server-server slouží ke kontrole zda by měla být zpráva specifikovaná pomocí ID zaslána na server s použitím příkazu TAKETHIS.

GROUP – slouží pro nastavení ukazatele skupiny (= výběr skupiny) a pro získání podrobných informací o skupině – počtu zpráv v ní, číslo první a číslo poslední dostupné zprávy

HEAD – chová se stejně jako příkaz ARTICLE. Jediný rozdíl je v tom že nevrátí celý text zprávy ale pouze jeho hlavičky.

HELP – vypíše seznam příkazů dostupných na serveru včetně jejich parametrů. Na různých serverech mohou být vráceny různé hodnoty, ne každý server totiž podporuje úplně všechny rozšířené NNTP příkazy (RFC 2980).

IHAVE – Příkaz IHAVE slouží k informování serveru že klient má k dispozici zprávu jejíž ID je předáno v parametru <messageid>. Pokud server tuto zprávu nemá a chce ji, bude vrácen kód 345, musíme pak zavést zprávu tak jako pomocí příkazu POST. Pokud je ale vrácen kód 435, znamená to že server už takovou zprávu má nebo ji nechce.

LAST – nastaví ukazatel zpráv na zprávu předcházející právě čtené zprávě. Jako odpověď server vrátí pořadové číslo této zprávy a její ID. Text zprávy zaslán v odpovědi není.

LIST – vrací seznam skupin dostupných na serveru. Původní verze příkazu, tak jak byla definována v RFC 977 nepřebírá žádné parametry. Nová verze podle RFC 2980 může používat několik modifikátorů (ACTIVE, ACTIVE.TIMES, DISTRIBUTIONS, DISTRIB.PATS, NEWSGROUPS, OVERVIEW.FMT, SUBSCRIPTIONS)

LIST ACTIVE - pokud je tento příkaz použit bez argumentu, vrací identický výsledek jako původní LIST. Jako volitelný argument můžeme použít masku (například

alt.binaries.*, comp.pro*,..). Příkaz poté vrátí pouze skupiny, které vyhovují této masce. V masce jde použít zástupný znak *, nelze použít regulérní výrazy.

LIST ACTIVE.TIMES - pomocí tohoto příkazu je na některých serverech možno získat informace o tom kdo a kdy založil kterou diskusní skupinu. Výstup má tři pole: název skupiny, čas který uplynul od založení skupiny (v sekundách od 1.1.1970) a emailovou adresu entity, která skupinu založila.

LIST DISTRIBUTIONS – některé servery uchovávají takzvaný distribution soubor. Obsahuje podrobnější informace o hlavičce Distribution, která je volitelnou součástí zpráv. Příkaz LIST DISTRIBUIONS vrátí informace obsažené v tomto souboru. Tyto hlavičky byly zavedeny s úmyslem umožnit odesilateli zprávy ovlivnit geografické území na které se zpráva má šířit (například zpráva s hlavičkou Distribution: world, !us by se měla šířit na všechny servery na světě kromě serverů v USA). Tento mechanismus se nicméně příliš velkého rozšíření nedočkal, jakékoliv omezování v podstatě odporuje principům na kterých byl Usenet vybudován.

LIST DISTRIB.PATS – některé servery uchovávají takzvaný distribution.pats soubor.

LIST NEWSGROUPS – některé servery uchovávají takzvaný newsgroups soubor. Ten obsahuje názvy jednotlivých skupin které server poskytuje a také krátký textový popis ke každé skupině.

LIST OVERVIEW.FMT – před spuštěním tohoto příkazu musí být vybrána nějaká diskusní skupina. Overview.fmt je soubor, který obsahuje všechny typy hlaviček které se ve zprávách v dané skupině vyskytují.

LIST SUBSCRIPTIONS – příkaz vrátí výchozí seznam předplacených skupin pro nově zaregistrované uživatele.

LISTGROUP - vypíše všechny pořadová čísla zpráv dostupných ve skupině specifikované v argumentu příkazu. Není-li zadán argument, vypíše pořadová čísla právě vybrané skupiny.

MODE READER - připojená entita tímto indikuje serveru že se k němu připojil klient, nikoliv server. Tento příkaz dělá v podstatě to samé jako příkaz SLAVE z původní normy RFC 977, který však byl zřídkakdy implementován.

MODE STREAM - při připojení typu server - server si server který se připojuje tímto příkazem může vyžádat přepnutí do proudového typu spojení (oproti u NNTP tradičnímu stavovému). Tento příkaz je nutno použít před použitím příkazů CHECK a TAKETHIS.

NEWSROUPS – vrátí seznam nově vytvořených skupin ve specifikovaném časovém rámci. Z principu může novou diskusní skupinu vytvořit kterýkoliv uživatel. Jediné omezení je lokální politika serveru na kterém má uživatel účet – záleží na správci zda a komu vytváření nových skupin povolí.

NEWNEWS – vrátí seznam nových zpráv ve vybrané skupině/skupinách ve specifikovaném časovém rámci.

NEXT – stáhne následující zprávu ve skupině

POST – slouží pro odesílání zprávy na server. Po zadání tohoto příkazu musíme zadat povinné hlavičky zprávy a poté samotné tělo zprávy.

QUIT – ukončí práci se serverem

SLAVE – dává serveru najevo že se k němu připojil jiný server (například za účelem výměny zpráv) a ne klient. Tento příkaz byl vytvořen proto aby se vnitřně dala rozlišit komunikace klientů od komunikace mezi servery. Serveru tak například může být přidělena vyšší priorita apod.

STAT – funguje podobně jako příkaz ARTICLE s tím rozdílem že nevrací žádný text zprávy. Pokud tento příkaz použijeme ve formě ve které se zadává pořadové číslo zprávy, slouží nám k nastavení ukazatele zprávy ve skupině a k tomu že nám vrátí její ID. Použití příkazu ve formě kdy mu předáme ID zprávy nemá žádné praktické využití.

TAKETHIS - v případě že je komunikace mezi servery nastavena na proudový typ spojení, slouží TAKETHIS k zasílání zpráv na server. Rozdíl oproti tradičnímu typu spojení je v tom že ihned po zadání příkazu TAKETHIS je celá zpráva (hlavičky i tělo zprávy) zaslána ihned bez toho aby se čekalo na potvrzující kódy.

XGTITLE - používá se k získání popisu specifické diskusní skupiny zadané v argumentu.

XHDR - vrátí specifikované hlavičky ze specifikovaných zpráv. Vyžadovaný parametr je název jedné hlavičky (například subject)

XINDEX - vrátí indexový soubor. Vyžadovaný parametr je název skupiny. Tento příkaz se dnes již prakticky nepoužívá, indexový soubor byl nahrazen rozšířenějším formátem NOV (news overview).

XOVER - vrátí informace o zprávách v dané skupině uložené v databázi overview. Tato databáze obvykle obsahuje nejdůležitější hlavičky. Argument příkazu XOVER je rozsah zpráv pro které se má výpis z databáze provést, zadaný ve formátu lokální číslo zprávy - lokální číslo zprávy.

XREPLIC - tento příkaz umožňuje přesně duplikovat strukturu poskytovaných souborů z jednoho serveru na druhý. Pracuje podobně jako příkaz IHAVE specifikovaný v RFC 977, používají se i stejné odpovídací kódy. Tento příkaz bylo nicméně vhodné použít pouze tehdy, pokud server, který chtěl duplikovat strukturu od jiného serveru přebíral svůj feed pouze od tohoto jediného serveru. Novější verze INN (nejrozšířenější NNTP server) jej už proto nepodporují a k zajištění identické kopie feedu používá modernější mechanismy (pomocí hlavičky Xref).

2.3 Návratové odpovědi

Existují dvě části návratové odpovědi protokolu– textová část a stavové kódy.

2.3.1 Textová část odpovědi

Textová odpověď je poslána výhradně až poté co byl zaslán stavový kód, který určuje co je obsahem následující textové odpovědi. Textová odpověď je zaslána jako série řádků. Každý řádek je ukončený dvojicí znaků CR-LF. Konec textové odpovědi je značen řádkem s jediným znakem tečky ‘.‘.

2.3.2 Stavové kódy

Stavový kód je tříciferné číslo zaslané serverem, které indikuje jaký druh reakce nastal na poslední příkaz zaslaný klientem. První pozice kódu určuje obecnou kategorii, druhá určuje konkrétní kategorii a třetí konkrétní kód. Konkrétní podoba jednotlivých návratových kódů je přesně vymezena u každého příkazu, viz. dále.

Významy kódů na první pozici:

- 1xx – informace, která může být ignorována
- 2xx – příkaz úspěšně ukončen
- 3xx – pokračujte v zadávání dat (u víceřádkových příkazů)
- 4xx – správný příkaz nemohl být proveden
- 5xx – nesprávný příkaz

Významy kódů na druhé pozici:

- x0x – spojení navázáno, server připraven a jiné obecné stavové hlášky
- x1x – zvolení diskusní skupiny
- x2x – volba zprávy ze skupiny
- x3x – funkce distribuce zpráv
- x4x – odesílání zpráv
- x8x – nestandardní příkazy
- x9x – data určená k debugování

Význam jednotlivých odpovědí protokolu NNTP

100 text nápovědy

190 ladící výstup (také 191 až 199)

200 server připraven – zasílání vlastních příspěvků je zde povoleno

201 server připraven – zasílání vlastních příspěvků zde není povoleno

202 slave status potvrzen

205 zavírám spojení

211 n f l s skupina vybrána

215 následuje výpis dostupných skupin

220 n <a> zpráva obdržena – následují hlavičky a tělo zprávy

221 n <a> zpráva obdržena – následují hlavičky zprávy

222 n <a> zpráva obdržena – následuje tělo zprávy

223 n <a> zpráva obdržena – text si vyžádejte zvlášť

231 seznam nově vytvořených skupin následuje

235 zpráva přenesena v pořádku

240 zpráva přijata v pořádku

335 zadejte zprávu která má být přenesena. Ukončete sekvencí “\r\n.\r\n“

340 zadejte zprávu která má být poslána. Ukončete ji sekvencí “\r\n.\r\n“

400 služba přerušena

411 skupina neexistuje

412 není vybrána žádná skupina

420 není vybrána žádná zpráva ve skupině

421 v této skupině není žádná textová zpráva

422 v této skupině není žádná starší (předchozí) zpráva

423 zpráva s tímto požadovaným číslem nebyla nalezena

430 žádná taková zpráva nebyla nalezena

435 tuto zprávu nechci – neposílejte ji

436 přenos selhal – zkuste to později

437 přijetí zprávy zamítnuto – nezkoušejte to později

440 zasílání zpráv není povoleno

441 zaslání zprávy selhalo

500 příkaz nebyl rozpoznán

501 chyba v syntaxi příkazu

502 omezení přístupových práv nebo nepovolený přístup k prostředku

503 porucha programu – příkaz nebyl vykonán

2.4 Hlavičky zprávy

Každá zpráva se skládá z hlaviček a z těla zprávy. Pokud chce klient připojit ke zprávě binární přílohu, je binární příloha zakódována některým z formátů k tomu určených a připojena před konec těla zprávy. Hlavičky nejsou součástí protokolu NNTP. Jsou definovány samostatně - v dokumentu RFC 1036. Tento dokument definuje řadu standardních

hlaviček. Norma však umožňuje flexibilně vytvářet vlastní hlavičky. Všechny takto „uživatelsky“ vytvořené hlavičky musí začínat předponou „X-„. [9]

Při zasílání příspěvku na server jsou povinné pouze 3 hlavičky - From, Newsgroups, Subject. Tyto hlavičky musí zadat klient při zasílání zprávy. Čtvrtá hlavička, která nesmí chybět u žádné zprávy je Message-ID. Message-ID je jednoznačně platný identifikátor který je jedinečný pro každou zprávu na světě. Tuto hlavičku sice klient může zadat sám ručně, je ale vhodnější když to neudělá. V tom případě ji totiž doplní automaticky server, na který je příspěvek zaslán.

Standardní hlavičky jsou tyto:

FROM: emailová adrese odesílatele zprávy. Povinná hlavička.

DATE: datum a čas kdy byla zpráva původně zaslána na server. Formát Wdy DD Mon HH:MM:SS YYYY

NEWSGROUPS: seznam diskusních skupin do kterých je zpráva odesílána. Může být zadáno i více než jedna skupina, každá položka je v tomto případě oddělena čárkou. Povinná hlavička.

SUBJECT: krátký popis zprávy - záhlaví. Povinná hlavička.

MESSAGE-ID: globálně jedinečný identifikátor. Při zasílání zprávy jej automaticky vytvoří a ke zprávě přidá server, klient ale může tuto hlavičku zadat i ručně. Formát je <unikátnířetězec@plné.doménové.jméno>. Hlavička je povinná ale pokud není zadána, vyplňuje ji automaticky server.

PATH: zde je zaznamenán každý server přes který zpráva při své cestě od odesilatele ke čtenáři prošla. Každý server po cestě přidá do této hlavičky svůj název, na začátek řetězce cesty. Původní server je tedy uveden nejvíce napravo. Jednotlivé názvy jsou odděleny vykřičníkem nebo jiným oddělovacím znakem kromě tečky. Název může být lokální jméno serveru nebo plné doménové jméno.

REPLY-TO: emailová adresa na kterou chce odesílatel obdržet případné reakce na svůj příspěvek.

SENDER: Označení entity (serveru nebo brány), odpovědné za odeslání příspěvku do Usenetu.

FOLLOWUP-TO: seznam skupin do kterých má být odeslána případná reakce na příspěvek.

EXPIRES: hlavička navrhuje datum po jehož uplynutí vyprší platnost zprávy (například pozvánka na konferenci). Servery tuto hlavičku nicméně mohou ignorovat.

REFERENCES: hlavička obsahuje message-ID všech zpráv na které tato zpráva reaguje. Je to mechanismus nutný k tomu aby mohlo jít vysledovat jednotlivá „vlákna“ diskuse. Tento příkaz by také měl vygenerovat takovou hlavičku Subject, která je stejná jako původní hlavička na kterou odpovídáme - s tím rozdílem, že na začátek řetězce jsou přidány čtyři znaky „Re: „.

CONTROL: pokud zpráva obsahuje tuto hlavičku, jedná se o řídící zprávu. Taková zpráva není určena k tomu aby byla zaslána jako příspěvek do nějaké skupiny, slouží jako mechanismus pro mazání zpráv, vytváření nových skupin, mazání „mrtvých“ skupin. Například pro smazání vlastní zprávy stačí zaslat zprávu s hlavičkou „Control: cancel <message-ID>“ (musí být přítomny i povinné tři hlavičky). Tělo zprávy se neuvádí, stačí ukončující sekvence - jeden prázdný řádek a tečka na následujícím řádku. Vytvoření nové moderované skupiny: „Control newgroup aaa.bbb.nova.skupina moderated“. Smazání existující skupiny: „Control rmgroup alt.test“. Control zprávy se šíří úplně stejně jako každé jiné zprávy na Usenetu. Problémem tohoto mechanismu je, že mazat zprávy a skupiny může úplně každý. Není vyžadována žádná autentizace (například že by skupinu mohl smazat pouze její zakladatel nebo zprávu její odesílatel). Protože s takovým otevřeným přístupem by k poničení celého Usenetu stačil pouze jediný nedisciplinovaný „uživatel“ s jednoduchým skriptem, podléhají tyto zprávy kontrole správců serverů. Toto sice není v žádné normě týkající se Usenetu, nicméně je to nutná sebezáchovná praxe. Zakládání nových skupin je povoleno většinou všude a ani s mazáním zpráv (třebas cizích) nebývá problém. Filtraci administrátorů ale přísně podléhají příkazy ke smazání skupiny. Servery jsou většinou nastaveny aby takovou zprávu nepustili do oběhu dříve než si ji prohlédne administrátor a v případě její oprávněnosti její vypuštění osobně povolí.

DISTRIBUTION: tento řádek ovlivňuje distribuční šíření zprávy z geografického hlediska. Cílová skupina zprávy je stále specifikována pomocí povinné hlavičky newsgroups, nicméně distribution by měla omezit její šíření pouze na servery nacháze-

jící se v dané geografické oblasti. Například zpráva „Distribution: nj, ny“ by měla patřit do oblasti New Jersey a New Yorku. Tento mechanismus má sloužit k tvorbě lokálních diskusních skupin, jejichž globální šíření by bylo zbytečné (například skupina pro nabídku a poptávku ojetých aut). Servery tuto hlavičku nicméně nemusí vůbec respektovat.

ORGANIZATION: obsahuje krátký popis organizace do které spadá odesilatel nebo počítače ze kterého je zpráva odeslána.

KEYWORDS: klíčová slova vztahující se k obsahu zprávy. Účelem je napomoci klientům při hledání konkrétního obsahu.

SUMMARY: stručné shrnutí obsahu zprávy.

APPROVED: tato hlavička je vyžadována pro kteroukoliv zprávu zasílanou do moderované skupiny. Moderované skupiny jsou takové, do kterých nejdou volně zasílat zprávy. Čtení zpráv v moderovaných skupinách ale nijak omezené není. Každá zpráva zaslaná do takové skupiny nejprve dojde na email moderátora, který se rozhodne zda k ní připojí hlavičku approved a pak ji přepošle do dané skupiny. Tento mechanismus je nicméně vytvořen značně nedomyšleně. Hlavičku approved totiž může do své zprávy přidat ručně kterýkoliv odesílatel a tím vlastně moderátora obejít. Žádná kontrola zda hlavičku approved do zprávy přidal moderátor se neprovádí (maximálně některé servery kontrolují adresu odesílatele s adresou moderátora dané skupiny, ale tuto hlavičku lze samozřejmě také libovolně pozměnit - třeba právě na adresu moderátora skupiny).

LINES: počet řádků zasílané zprávy. Obyčejně přidává server.

XREF: jméno hostitele na kterém je zpráva nyní uložena následována seznamem všech ostatních skupin do kterých byla zpráva také zaslána + pořadové číslo této zprávy v každé z vypsaných skupin. Kvůli lokálnímu charakteru se tato hlavička nepřenáší.

2.5 Tělo zprávy

Tělo zprávy začíná klíčovým slovem „body“ na samostatném řádku. Od dalšího řádku pak následuje samotná zpráva. Může ji tvořit jakákoliv posloupnost textových znaků.

Tělo zprávy je ukončeno sekvencí “\r\n.\r\n“ (neboli prázdný řádek, řádek s tečkou, prázdný řádek).

2.6 Binární přílohy

Mnoho diskusních skupin na Usenetu je založeno za účelem šíření binárních příloh. Binární přílohy jsou vše co není text zprávy nebo hlavička – tedy jakékoliv přiložené soubory (dokumenty, obrázky, zvuky, filmy, komprimované soubory, programy,..). Binární přílohy jsou tedy analogií emailových příloh. Usenet byl nicméně původně navržen pouze pro šíření textových informací (stejně jako email). Bylo tedy nutné nalézt způsob jak přes textový protokol zasílat netextové informace. Čistý ASCII text je svým charakterem mezi binárními daty poměrně výjimečná struktura, a to protože k zobrazení všech „strojopisných“ znaků je potřeba pouze 6 bitů (64 kombinací). Binární data naproti tomu pro zobrazení jednoho znaku – byte – potřebují 8 bitů. Z hlediska textových protokolů pak mají některé z těchto kombinací bitů (například 0x0, 0xd, 0xa a další) specifický význam – jsou to řídící znaky. Jejich přímý přenos v textu zprávy by tedy nenapravitelně narušil komunikaci mezi klientem a serverem.

Řešení přinesly standardy UUEncode, MIME/base64 a yEnc. Formát UUEncode je nejstarší a nejpoužívanější. Zkratka znamená „unix to unix encoding“, má tedy svůj původ v Unixu. MIME znamená Multipurpose Internet Mail Extensions neboli víceúčelové emailové rozšíření. Base64 je jedna z metod kterou MIME definuje. MIME je specifikován v dokumentech RFC 2045, 2046, 2047. Formát yEnc je nejmladší. Jeho výhodou jsou velmi malé režie při převodu – objem přenesených dat naroste pouze o 1-2% oproti 33%-40% u uuencode a MIME. Kromě toho yEnc přímo podporuje CRC hashování, jež slouží k ověření zda nebyl obsah zprávy při přenosu změněn.

Znaková sada uuencode se skládá z těchto znaků:

```
!\"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[]^_
```

Znaková sada base64 se skládá z těchto znaků:

```
sABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
```

Program vytvořený v této práci implementuje kódování UUEncode, protože je nejrozšířenější.

2.6.1 UUEncode

UUEncode je algoritmus který slouží k posílání binárních dat v rámci textových zpráv. Jako zdrojová data slouží jakýkoliv binární soubor. Výstupem je posloupnost znaků z podmnožiny ASCII znaků – znaky s pořadovým číslem 32 (mezera) až 96 (_).

Výstup je vložen do těla zprávy, většinou na konec. Část zprávy, ve které je uložen soubor zakódovaný uuencode začíná novým řádkem, na kterém je řetězec ve formátu

begin <mód> <jméno souboru>

Begin je povinné klíčové slovo. Proměnná mód označuje přístupová práva pro čtení/zápis/spuštění (většinou 644) a proměnná soubor jméno souboru ze kterého zakódovaná data pochází. Konec zakódovaných dat je označen novým řádkem obsahujícím znak apostrof ‘ a ještě jedním novým řádkem obsahujícím řetězec „end“.

Algoritmus zakódování dat je následující: Ze zdrojového souboru dat načteme vždy 3 byty najednou. V bitovém vyjádření se jedná o 24 bitů. Těchto 24 bitů rozdělíme na čtvrtiny - čtyři šestice bitů. Ke každé šestici přidáme na nejvyšší dvě bitové pozice dvojkové nuly. Tím dostaneme 4 plné byty. Každý byte pro nás vyjadřuje ve dvojkové soustavě číslo z intervalu 0-63 (vyšší toto číslo být nemůže protože na 8. a 7. pozici jsou nuly dosazené v minulém kroku). Nyní ke každému z těchto čísel přičteme číslo 32. Tím dostaneme rozsah který odpovídá požadovaným ASCII hodnotám. Všechny znaky, které takto dostaneme jsou „strojopisné“ a jdou bez problémů přenášet textovým protokolem. V tomto procesu existuje jediná výjimka. Pokud je výsledný zakódovaný znak mezera (ASCII 032), je nahrazen znakem ‘ (ASCII 096) z důvodu že některé servery při přenosu zpráv ořezávají „nepotřebné“ vícenásobné mezery a tím celý zakódovaný soubor naruší.

Základní princip UUEncode je tím popsán. Celý proces převodu binárního souboru do textový zprávy však ještě nekončí. Každá skupina 60 výstupních bytů (převedená z původních 45 bytů na vstupu) je považována za samostatný řádek uvedený znakem ‘M’. Proč znak M? Jeho ordinální číslo v ASCII tabulce je 77. Pokud od 77 odečteme 32 (číslo které přičítáme ke všem bytům výstupu), dostaneme 45 což je právě počet bytů na vstupu připadajících na jeden řádek. M na začátku řádku nám tedy uvozuje kolik znaků vstupu máme na řádku vlastně zakódovaných. Co se týká posledního řádku výstupu, pokud neobsahuje plných 60 znaků (tedy pokud vstup není dělitelný číslem 45, což nastane zhruba v 98% případů), neuvozuje se znakem M, ale znakem ASCII tabulky, který získáme

z tohoto vztahu: počet zbývajících bytů na vstupu + 32. Poté následuje řádek s povinným apostrofem ‘ a ještě jeden řádek s povinným řetězcem end.

Příklad UUEncode [12]

Vstupní znaky - jedna trojice	c	a	t
Desítkové vyjádření	67	97	116
Dvojkové vyjádření	010000 11	0110 0001	01 110100
	↓	↓ ↓	↓
Výstupní znaky - čtveřice	010000	110110 000101	110100
Přidání nul	0001 0000	0011 0110 0000 0101	0011 0100
Desítkové vyjádření	16	54 5	52
+32	48	86 37	84
Výstupní řetězec	0	V %	T

ASCII řetězec „cat“ má tedy v UUEncode vyjádření podobu „0V%T“. Řetězec „cat“ by samozřejmě šel v textovém protokolu přenést rovnou bez jakýchkoliv úprav, pro názornost příkladu by ale nebylo vhodné používat jiné než textové znaky. Pokud celý soubor tvořil pouze tento řetězec, byl by do NNTP zprávy vložen s následujícími povinnými řetězci takto:

```
begin 644 cat.txt
#0V%T
`
end
```

Znak ‘#’ má v ASCII tabulce pořadové číslo 35 = 32 + 3 byty na posledním řádku vstupu.

2.7 Ukázková komunikace

2.7.1 Čtení zprávy

Ukázkový výpis komunikace s NNTP serverem pomocí standardního programu TELNET (tučně jsou zvýrazněny vstupy z klávesnice, zbytek textu je odpověď severu):

```
c:\>telnet news.telecom.cz 119
200 localhost InterNetNews NNRP server INN 2.3.5 ready (posting ok).
mode reader
```

200 localhost InterNetNews NNRP server INN 2.3.5 ready (posting ok).

list

215 Newsgroups in form "group high low flags".

aaa.inu-chan 0000000000 0000000001 y

ab.arnet 0000000000 0000000001 y

ab.general 0000000000 0000000001 y

ab.jobs 0000000000 0000000001 y

…

(výpis seznamu zkrácen, má více než 44000 položek)

…

alt.israel.test 0000000000 0000000001 y

alt.israel.t3st 0000000000 0000000001 y

alt.fr.capitalisme.la.revanche 0000000000 0000000001 y

alt.test.shitcannon 0000000000 0000000001 y

.

group comp.ai.neural-nets

211 126 1486 1612 comp.ai.neural-nets

article 1595

220 1595 <1148024011.361163.136580@j33g2000cwa.googlegroups.com> article

Path: localhost!bnewsinpeer00.bru.ops.eu.uu.net!emea.uu.net!newsfeed.freenet.de!news.germany.com!postnews.google.com!j33g2000cwa.googlegroups.com!not-for-mail

From: "classic" <john.atwell.moody@gmail.com>

Newsgroups: comp.ai.neural-nets

Subject: Proof that neural nets work

Date: 19 May 2006 00:33:31 -0700

Organization: http://groups.google.com

Lines: 12

Message-ID: <1148024011.361163.136580@j33g2000cwa.googlegroups.com>

NNTP-Posting-Host: 82.31.3.138

Mime-Version: 1.0

Content-Type: text/plain; charset="iso-8859-1"

X-Trace: posting.google.com 1148024016 1328 127.0.0.1 (19 May 2006 07:33:36 GMT)

X-Complaints-To: groups-abuse@google.com

NNTP-Posting-Date: Fri, 19 May 2006 07:33:36 +0000 (UTC)

User-Agent: G2/0.2

X-HTTP-UserAgent: Mozilla/4.0 (compatible; MSIE 6.0; Windows 98),gzip(gfe),gzip(gfe)

Complaints-To: groups-abuse@google.com

```
Injection-Info: j33g2000cwa.googlegroups.com; posting-host=82.31.3.138; posting-
account=JWLXLQ0AAABsyG63OZNJyjEYUrlVwdQK
Xref: localhost comp.ai.neural-nets:1595

Two important announcments. 1. The text file
http://www.goldengem.co.uk/proof.txt contains complicated combinations
of trig functions. Also the website has the a free neural net  (same
site, omit /proof.txt). Point it at the data and you can really see how
well it predicts these complicated functions. This shows, if there is a
pattern, a neural net will see it.

2. On 17 May the configuration of GoldenGem was enlarged, not a lot,
but it has made a tremendous difference. The photos are proof of that,
and you can try it yourself. If you have a reg number or are during
trial interval, you are welcome to download the improved version

.

quit

205 .
Připojení k hostiteli bylo ztraceno.
c:\>_
```

Ve výpisu došlo pomocí programu telnet k připojení na veřejný server nttp://news.telecom.cz, běžící na portu 119. Kódem 200 bylo oznámeno úspěšné připojení, zároveň server v hlavičce dává na vědomí že je zde povolené zasílání příspěvků – posting ok. Příkaz mode reader je u naprosté většiny současných konfigurací serverů zbytečné zadávat, servery jaksi implicitně očekávají že se připojil klient. Podle normy sem však patří. Příkaz list provede výpis všech skupin dostupných na serveru. Jak je vidět je jich zde vedeno hodně, na druhou stranu jsou až na výjimky zcela prázdné – jak ukazuje první číslo za názvem každé skupiny, většinou neuchovávají ani jeden příspěvek. Jedna z mála „obydlených“ skupin je comp.ai.neural-nets – je zvolena příkazem group comp.ai.neural-nets. Podle názvu se jedná o skupinu zabývající se kategoriemi počítače/umělá inteligence/neuronové sítě. Zpráva, která byla vypsána, má v záhlaví uvedeno „Důkaz že neuronové sítě fungují“.

2.7.2 Zaslání zprávy

Jak již bylo zmíněno v části zabývající se příkazy protokolu, k zaslání vlastní zprávy na server slouží příkaz POST. Po připojení a autentizování na serveru, který má zasílání zpráv povoleno, je postup následující: na server je zaslán příkaz POST. Pokud serve odpoví „340 Ok, recommended ID <aaaabbbb0123@news.adresa.cz>" mohou být zadány hlavičky a tělo zprávy. Jak je vidět v odpovědi, server sám navrhne hlavičku Message-ID (v tomto případě smyšlenou <aaaabbbb0123@news.adresa.cz>"), pokud by si zasílající nepřál zadat ji sám. Hlavičky zpráv jsou probírány v příslušné kapitole. Povinné pro každou zprávu jsou pouze 3: From, Newsgroup a Subject. Každá hlavička patří na samostatný řádek. Po zadání hlaviček následuje klíčové slovo „Body: „ (pozor na mezeru na dvojtečkou, některé servery ji vyžadují!) na samostatném řádku, poté prázdný řádek a poté již samotný text zprávy. Ukončení těla zprávy je oznámeno zadáním jednoho prázdného řádku následovaného jedním řádek obsahujícím pouze znak tečka '.'. Po odřádkování tohoto řádku se zpráva odešle. Pokud vše proběhlo bez komplikací, vrátí server odpověď ve tvaru „240 Article posted <aaaabbbb0123@news.adresa.cz>". Je-li vybrána stejná skupina do které byl článek zaslán, lze si jeho přijetí snadno ověřit příkazem „ARTICLE <aaaabbbb0123@news.adresa.cz>".

Výpis ukázkového zasílání zprávy na server:

```
c:\>telnet news.telecom.cz 119
200 localhost InterNetNews NNRP server INN 2.3.5 ready (posting ok).
mode reader
200 localhost InterNetNews NNRP server INN 2.3.5 ready (posting ok).
post
340 Ok, recommended ID <e68i5h$t1v2@localhost.localdomain>
from: filip.merhaut@centrum.cz
subject: test
newsgroups: alt.test
body:

Toto je testovaci zprava.

.

240 Article posted <e68i5h$t1v2@localhost.localdomain>
```

```
quit

205 .
Připojení k hostiteli bylo ztraceno.
c:\>_
```

Server, uvedený v tomto příkladu nevyžaduje autentizaci. Po připojení je tedy možno příkazem mode leader rovnou oznámit že se připojil klient a poté příkazem post spustit zasílání zprávy. Po zadání povinných hlaviček je zadán řádek body: a za ním krátký text zprávy, ukončený požadovanou sekvencí. Návratový kód 240 poté oznamuje že byla zpráva úspěšně odeslána na server. Příkazem quit je ukončena práce se serverem.

Jakýkoliv grafický klient pro práci s Usenetem musí vykonávat stejné sekvence příkazů jako v těchto příkladech. Rozdíl je pouze v tom, že běžný uživatel tyto příkazy nevidí, jsou skryty na grafickou nadstavbou programu, který za něj obslouží technické podrobnosti a povinné režie a umožní tak uživateli soustředit se pouze na samotné čtení a zasílání zpráv.

II. PRAKTICKÁ ČÁST

3 VÝVOJOVÉ PROSTŘEDKY

3.1 .NET Framework

K vývoji zadaného programu byla zvolena technologii .NET firmy Microsoft. Je to moderní platforma pro tvorbu programů. Mezi důležité prostředky této platformy patří objektové programování, bezpečný typový systém, automatická správa paměti, rozsáhlá skupina základních tříd poskytující tisíce funkcí z nejrůznějších oblastí, včetně snadné tvorby grafického uživatelského rozhraní v operačních systémech Windows a v neposlední řadě také podpora moderních technologií typu Unicode a XML. Při tvorbě tohoto programu byly využity třídy ve verzi .NET Framework 1.1. [3]

Při vytváření programu středního rozsahu jako je tento klient bylo rozumnou volbou zvolit takový vývojový prostředek, který je založen na principech objektového programování. Objektové programování umožňuje tvořit aplikace jako systém samostatných spolupracujících jednotek - objektů. Každý objekt přitom si přitom uchovává detaily své implementace uvnitř a navenek ostatním objektům poskytuje pouze svou funkcionalitu. Starší verze objektů tak mohou být bez obtíží nahrazeny novějšími - s lepším výkonem, lepšími algoritmy, opravenými chybami, apod. Samozřejmě za podmínky že se v nových verzích nezmění vnější rozhraní objektu. Spolu s dědičností a polymorfismem lze takto vytvářet snadno opravitelné a rozšiřitelné aplikace, což je důležité zejména u větších projektů. Jako podpůrný vývojový prostředek byl pro odlaďování a kontrolu síťového provozu použit program Packetyzer.

3.2 Jazyk C#

C# (vyslovováno anglicky jako C sharp) je vysokoúrovňový objektově orientovaný programovací jazyk, vyvinutý firmou Microsoft zároveň s platformou .NET. C# je potomkem jazyků rodiny C a jazyka Java. Je to výhradně objektově orientovaný jazyk. Využívá se převážně k tvorbě databázových programů, webových služeb a formulářových aplikací v operačním systému Windows. Firma Microsoft nabízí několik verzí překladače jazyka C#. Ta nejjednodušší, která k sestavení této aplikace plně postačuje, se nazývá Visual C# Express Edition a je k dispozici zdarma ke stažení.

4 ROZBOR APLIKACE

Jedním z prvních a také klíčových rozhodnutí při tvorbě každé aplikace je vhodné rozvržení struktury programu na základě dekompozice a analýzy řešeného problému. Nedostatek pozornosti věnované této fázi tvorby programu se později mnohonásobně vymstí v podobě slepých uliček, „záplatování“ špatné funkcionality, špatné rozšiřitelnosti a celkově chaotického vývoje programu. Promýšlení koncepce programu tedy byl věnován dostatek času.

4.1 Funkce programu

Funkce programu byly víceméně dány dopředu samotným protokolem NNTP a také zkušenostmi s běžnou prací s Usenetem. V každém programu je možné vymýšlet inovativní funkce, nejdůležitější ovšem je aby dobře zvládal to k čemu je primárně určen. Proto byl sestaven následující soupis funkcionality, kterou by zadaný program měl bezvýhradně poskytovat:

1. vést si seznam serverů včetně informací nutných k autorizaci přístupu
2. připojit se na vybraný server, stáhnout a zobrazit seznam skupin
3. u vybrané skupiny vypsat hlavičky všech dostupných zpráv v této skupině
4. přečíst vybranou zprávu s možností uložení přílohy zprávy do souboru
5. odpovědět na čtenou zprávu
6. zaslat vlastní zprávu na server (pokud má server povolený posting)

To vše v grafickém uživatelském rozhraní (GUI) MS Windows, s podporou více vláken (multithreading).

4.2 Použité třídy grafického uživatelského rozhraní GUI

Tvořit v operačním systému Windows grafické uživatelské rozhraní lze v podstatě dvěma způsoby: buď přímo voláním služeb WinAPI nebo použitím nadstavbových prostředků (knihoven různých výrobců, které ale rozhraní WinAPI stejně volají). Předností WinAPI je možnost dosáhnout velké výkonnosti výsledné aplikace. Nadstavbové prostředky jsou výhodné pro efektivnější a pohodlnější práci. WinAPI je poměrně složitý systém, v němž zabere velkou část programátorova úsilí obsluha nejrůznějších režijních úkonů. WinForms je nadstavba přímo od firmy Microsoft. Vývojářům pod OS Windows poskytuje nebývalý komfort. V této práci byla využita široká řada tříd WinForms ze jmenného pro-

storu System.Windows.Forms: Form, MainMenu, StatusBar, TextBox, ListView, ProgressBar, ContextMenu, ToolBar, Buton, PictureBox, Label, GroupBox. [4]

4.3 Multithreading

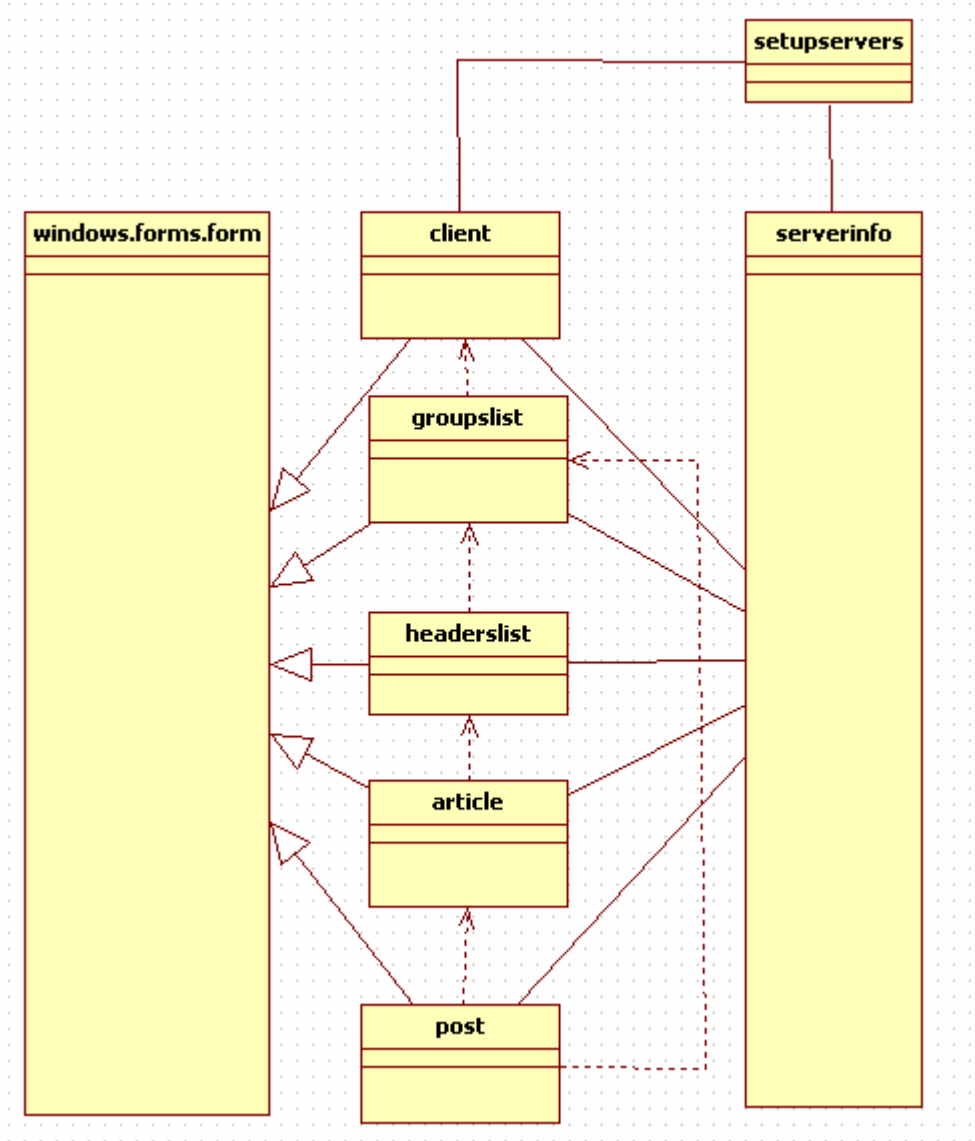
Pro použití konstrukce programu s více vlákny je potřeba mít důvod. Časté využití naleznou tam, kde je potřeba vykonávat výpočty nebo práci s daty na pozadí takovým způsobem, aby nebylo blokováno uživatelské rozhraní a uživatel tedy mohl dále pracovat s programem bez toho že by musel čekat na dokončení operace. Vlákna však nejsou všelék. Jejich špatné použití může aplikaci silně zkomplikovat, případně zhoršit její výkon. Důvod pro použití více vláken v tomto projetu představovala skutečnost, že AlephNews pracuje s velmi rozsáhlými seznamy dat. Je to dáno skutečností že počet skupin na jednom serveru se běžně pohybuje v řádech desetitisícíců. Počet zpráv v některých diskusních skupinách pak dokonce ve statisících, v několika extrémních případech až milionech kusů v časovém rámci který server uchovává (běžně jeden měsíc). Natažení seznamu skupin tak na některých serverech trvá až několik minut.

K práci s vlákny poskytuje .NET framework třídy ve jmenném prostoru System.Threading. Základní třídy jsou Thread a ThreadStart. Thread je objekt představující konkrétní vlákno. Jeho konstruktor vyžaduje objekt typu ThreadStart, který obsahuje základní informace nutné ke spuštění vlákna. Nejdůležitější z těchto informací je zástupce metody, která bude vláknu přiřazena – tedy po jeho nastartování bude spuštěna. Tento zástupce se předává do konstruktoru třídy ThreadStart. [5]

Jak již bylo zmíněno, při práci s vlákny je důležitá obezřetnost. Kromě správného spuštění musí mít programátor také zřeteli kdy je nutné vlákno ukončit. Vytvoříme-li totiž nové vlákno v dceřiném okně a toto okno někdy v průběhu běhu vlákna zavřeme, nemá to na běh vlákna žádný vliv – vlákno běží dál. Pokud tedy například stahujeme velmi dlouhý seznam diskusních skupin a zavřeme celou aplikaci, bude vlákno stahující seznam pracovat dále na pozadí a velmi tím zatěžovat procesor a síťový provoz. V aplikaci je proto u každého okna sledováno, které vlákna jsou v něm spuštěny a při případném zavření okna jsou tyto vlákna uzavírány metodou Abort().

5 TŘÍDY

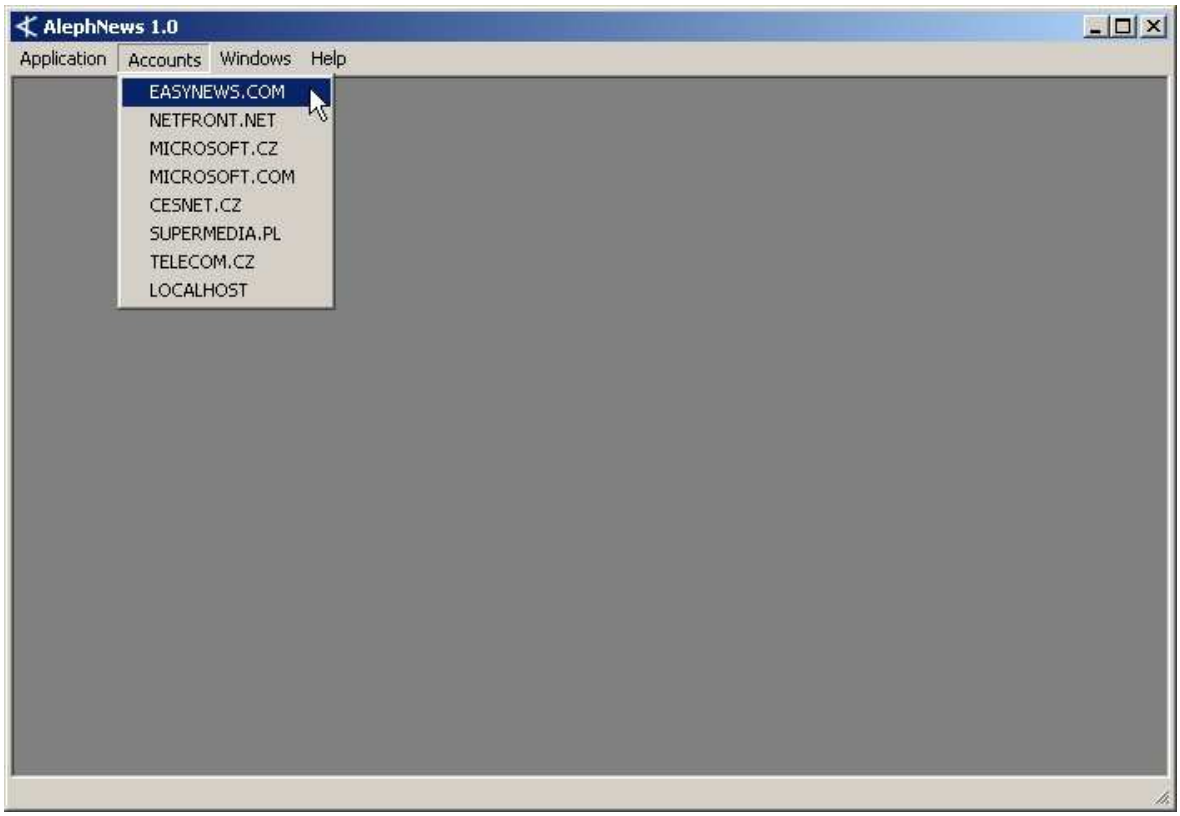
Výsledný program je tvořen množinou tříd, které vzájemně spolupracují tak, aby program plnil požadované funkce. Vstupním bodem je hlavní okno aplikace, třída Client, na ní závisí třída GroupsList se seznamem diskusních skupin na serveru. Na tu navazuje třída HeadersList, která obsluhuje seznam hlaviček zpráv v jedné skupině. Z obou předchozích tříd lze volat třídu Article, představující jednu zprávu. Ze předchozích tří tříd lze volat třídu Post, která poskytuje prostředky pro zaslání vlastní zprávy nebo odpovědi na server. Vše datově propojuje třída ServerInfo. Data pro její naplnění jsou obsluhovány třídou SetupServers. Na následujícím UML diagramu jsou uvedeny pouze hlavičky tříd, nejsou zde uvedeny členské proměnné ani metody jednotlivých tříd, neboť je jich příliš mnoho.



Obr. 2. UML class diagram aplikace

5.1 Client

Základní třída programu. Je jí předáno řízení po spuštění programu. Přes rozbalovací menu nabízí přístup ke všem funkcím. Především ale funguje jako MDI kontejner pro ostatní okna programu. Protože každé okno své funkce spouští ve vlastním vláknu, lze otevřít libovolný počet oken. Vytváří instance objektů SetupServers a GroupsList.

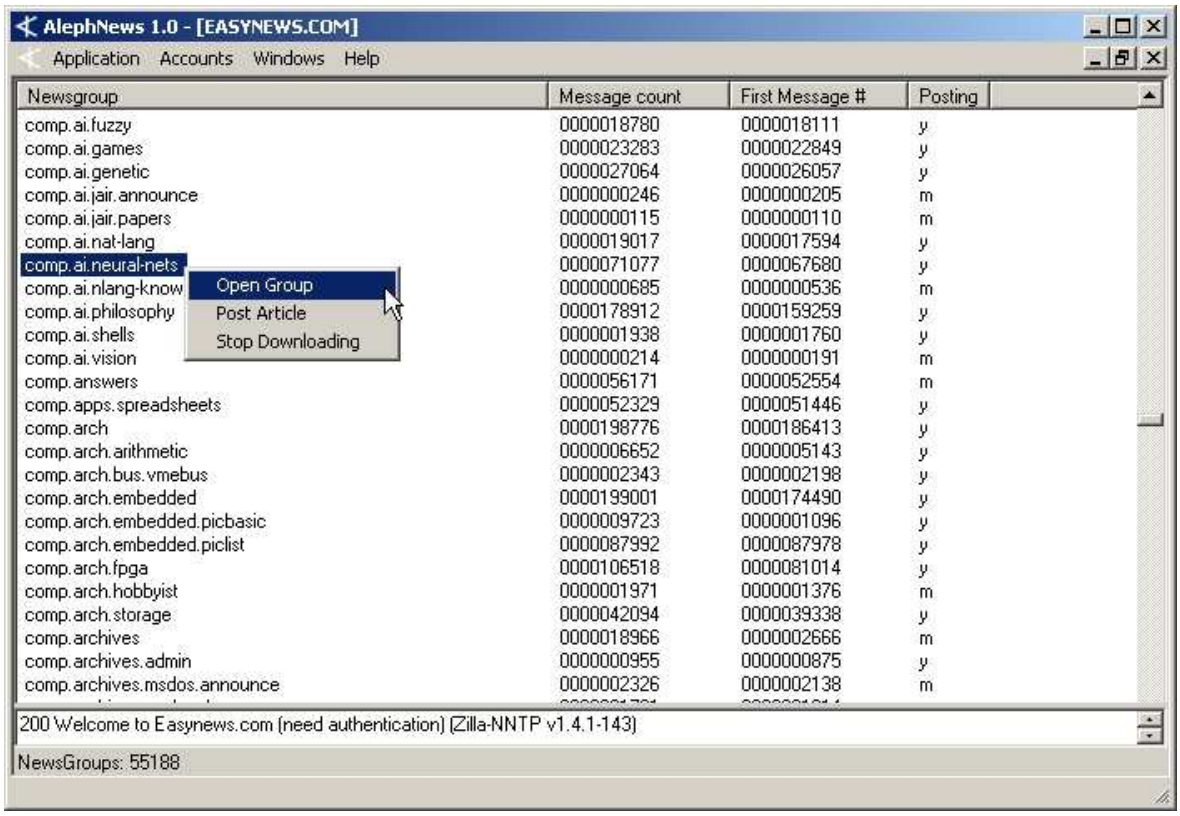


Obr. 3. Formulář třídy Client

5.2 GroupsList

Konstruktor třídy přebírá instanci třídy ServerInfo. Třída slouží k zobrazení seznamu diskusních skupin na serveru. Obsahuje metodu pro připojení na NNTP server a stažení seznamu skupin, které tento server odebírá. Tato metoda je spuštěna v samostatném vlákně, díky tomu je seznam skupin v při stahování průběžně aktualizován a ani při velmi rozsáhlém seznamu nemusí uživatel čekat na dokončení operace. Po kompletním stažení seznamu je možné jej setřídit podle jednotlivých sloupců. Při právě probíhajícím stahování seznamu není možno seznam třídit, tato funkce měla výrazně negativní vliv na výkon aplikace. Stahování je také možno v průběhu přerušit. Vybranou skupinu ze seznamu lze otevřít, tím dojde k vyvolání instance objektu HeadersList. Z tohoto seznamu skupin je také

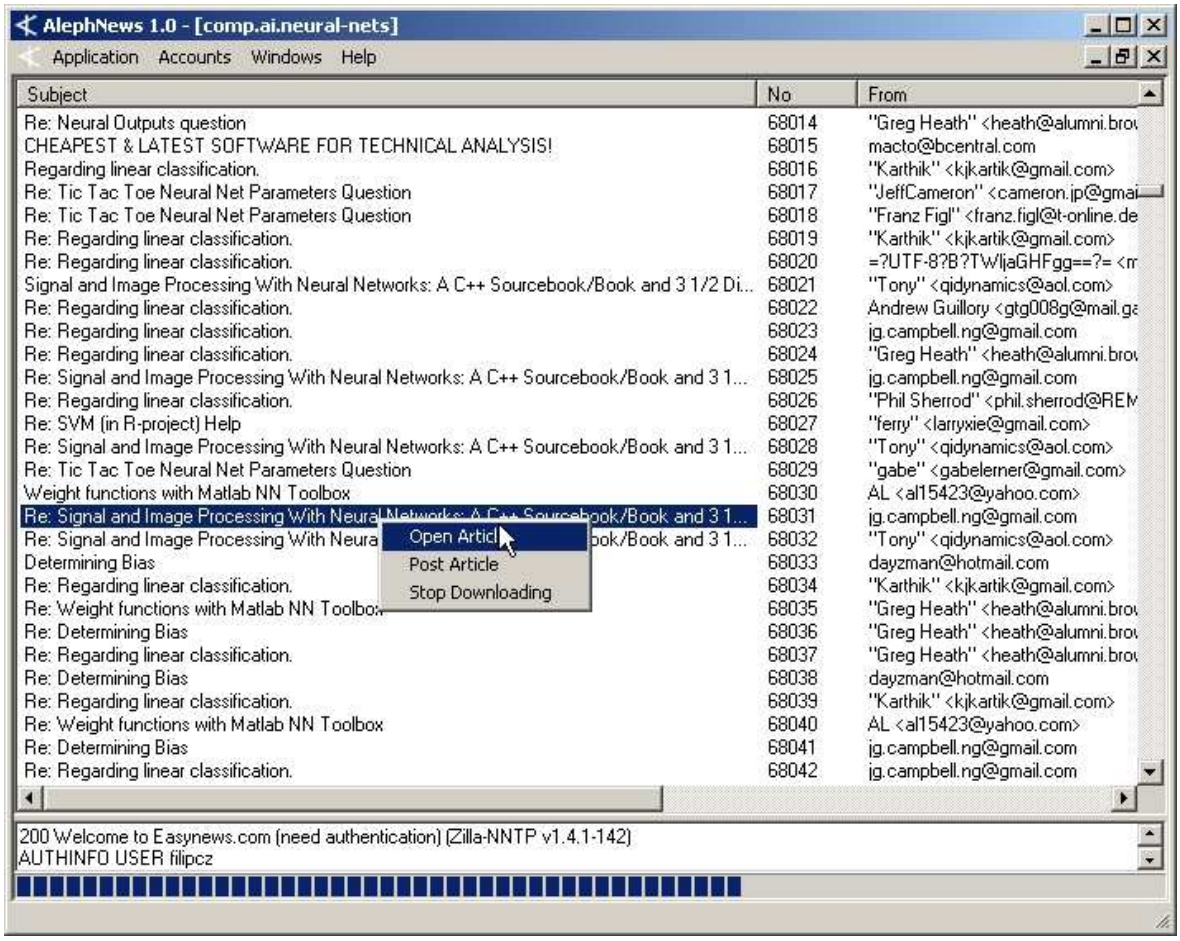
možno zaslat do diskusních skupin vlastní zprávu. Vybrání této volby přes kontextové menu vyvolá instanci třídy Post s předem vyplněnými údaji potřebnými k zaslání zprávy do vybrané skupiny.



Obr. 4. Formulář třídy GroupsList

5.3 Headerslist

Konstruktor třídy přebírá instanci třídy ServerInfo. Třída slouží k zobrazení seznamu příspěvků v diskusní skupině. Obsahuje metodu pro připojení na NNTP server a stažení seznamu zpráv ve skupině. Tato metoda je spuštěna v samostatném vlákně, díky tomu je seznam zpráv v při stahování průběžně aktualizován. Po kompletním stažení seznamu je možné jej setřídit podle jednotlivých sloupců. Při právě probíhajícím stahování seznamu není možno seznam třídit, tato funkce měla výrazně negativní vliv na výkon aplikace. Stahování je také možno v průběhu přerušit. Vybranou zprávu ze seznamu lze otevřít, tím dojde k vyvolání instance objektu Article. Z tohoto seznamu hlaviček je také možno zaslat do diskusní skupiny vlastní zprávu. Vybrání této volby přes kontextové menu vyvolá instanci třídy Post s předem vyplněnými údaji potřebnými k zaslání zprávy do této skupiny.



Obr. 5. Formulář třídy HeadersList

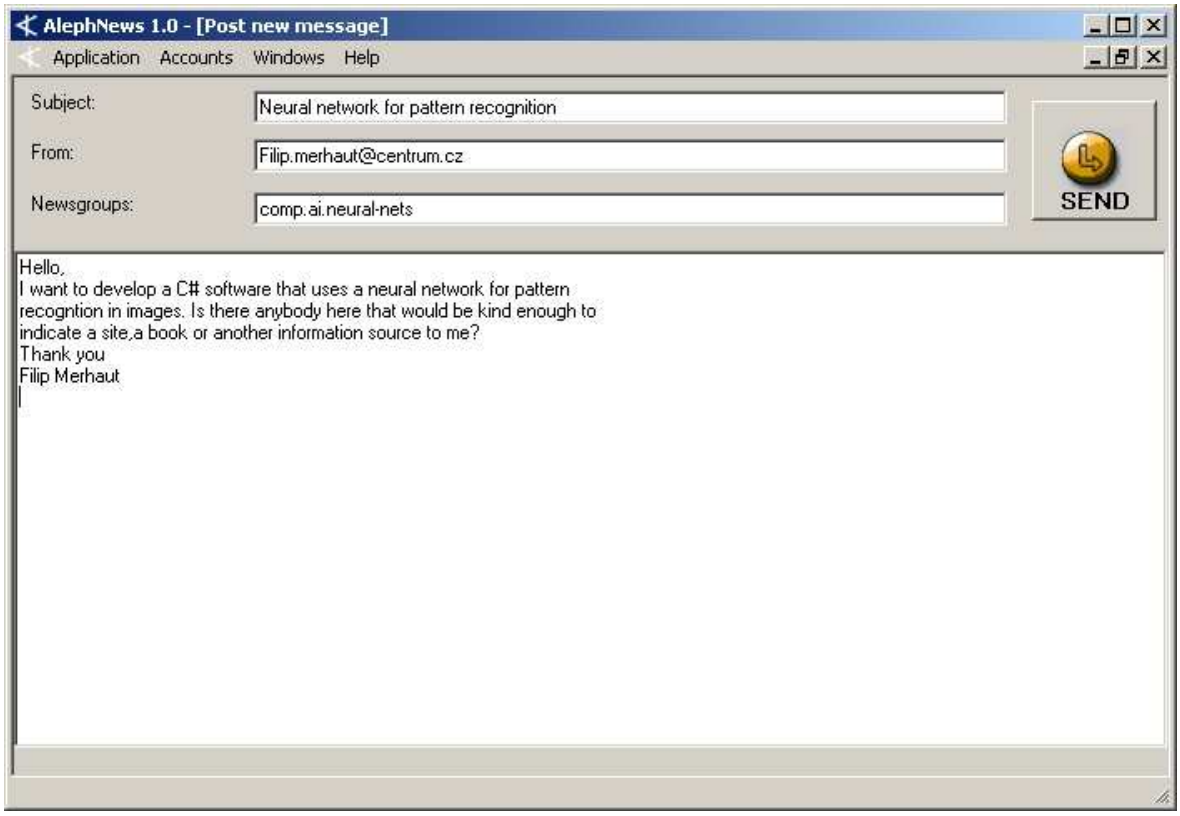
5.4 Article

Konstruktor třídy přebírá instanci třídy ServerInfo. Třída slouží k zobrazení konkrétní zprávy. Obsahuje metodu pro připojení na NNTP server a stažení zprávy specifikované v objektu typu ServerInfo. Tato metoda je spuštěna v samostatném vlákně, zpráva je tedy zobrazována průběžně. Stahování velmi velké zprávy je možno přerušit. Pro odpověď na daný článek je k dispozici je tlačítko Reply, které vyvolá instanci třídy Post s předem vyplněnými údaji potřebnými k zaslání navazující zprávy. Pokud zpráva obsahuje binární přílohu kódovanou pomocí standardu UUEncode, je tato příloha dekódována do podadresáře downloaded v pracovním adresáři aplikace. Uživatel si poté může pomocí dialogu „Save the atachment“ vybrat kam daný soubor uložit. Pokud je jako příloha připojen obrázek ve formátu bmp, jpg nebo gif, zobrazí se přímo v okně aplikace jako součást zprávy.

Obr. 6. Formulář třídy Article

5.5 Post

Tato třída má dva přetížené konstruktory. Jeden přebírá pouze objekt typu ServerInfo, druhý kromě něj i dva objekty typu string. Jedná se o formulář sloužící k odeslání vlastní zprávy nebo k odeslání odpovědi na existující zprávu. To kterou funkci vykoná záleží na tom se kterou verzí konstruktoru je třída vytvořena. Připojení na server a odeslání těla zprávy je v samostatném vláknu, aplikace tedy není blokována.



Obr. 7. Formulář třídy Post

5.6 Setupservers

Tato třída obsluhuje seznam serverů. Seznam je uložen k souboru servers.xml v adresáři aplikace. Tento formulář je modální, nelze se tedy přepnout zpět do aplikace dokud nebude tento formulář zavřen. Kdyby to tak nebylo, mohlo by dojít ke kolizi při které by se více procesů snažilo otevřít soubor servers.xml. Tato část programu vnitřně používá třídy ze jmenného prostoru System.XML.

Obr. 8. Formulář třídy SetupServers

5.7 Serverinfo

Tato třída slouží jako spojovací můstek mezi všemi třídami programu. Všude tam, kde je v ostatních třídách potřeba se připojit na konkrétní server jsou informace a adrese, portu, loginu, heslu apod. předávány právě přes instanci této třídy.

6 SYSTÉMOVÉ POŽADAVKY A KOMPILACE

6.1 Systémové požadavky

Základní podmínkou ke spuštění programu vytvořeného v této práci je přítomnost běhového prostředí .NET Framework verze 1.1 nebo novější v operačním systému. Systémové požadavky programu jsou dány požadavky tohoto běhového prostředí. Minimální požadavky umožňující spuštění aplikace jsou:

- Operační systém: Windows 98, Windows ME, Windows NT 4.0sp6a, Windows 2000, Windows XP, Windows 2003 (na všech těchto systémech je také nutné mít nainstalovány MS Internet Explorer 5.01 nebo novější, MS Windows Installer 2.0 nebo novější a .NET Framework 1.1 nebo novější).
- Procesor: Pentium 133 MHz nebo rychlejší
- Operační pamět: 128 MB nebo více

Je nutno poznamenat že na počítači těchto parametrů je sice možné program spustit, ale práce s ním by byla velmi pomalá.

6.2 Kompilace

Pro sestavení programu do spustitelné podoby je potřeba mít nainstalovaný .NET Framework alespoň verze 1.1. Tato distribuce je poměrně běžná na většině současných PC s operačním systémem Windows. Kromě běhového prostředí CLR, potřebného ke spuštění výsledné aplikace obsahuje i řádkový překladač jazyka C#, csc.exe. Sestavovat program této velikosti přes řádkové příkazy je možné, ale je to značně nepohodlné. Proto je lepší využít některé integrované vývojové prostředí z řady Visual Studio .NET. Díky použité verzi .NET Frameworku lze program přeložit ve všech verzích počínaje Visual Studio .NET 2003. Verze Visual Studio Express Edition je na stránkách firmy Microsoft k dispozici ke stažení zdarma.

ZÁVĚR

Výsledný program byl pojmenován AlephNews. Implementuje veškerou funkcionalitu potřebnou pro běžnou práci s Usenetem. Jeho interface je střídmý, ale funkční. Platforma .NET, na které je postaven, lehce limituje jeho výkon ve chvílích kdy je potřeba pracovat s extrémně rozsáhlými (desetitisíce, statisíce) seznamy diskusních skupin či příspěvků v nich. Takové skupiny se v dnešním Usenetu vyskytují, nicméně jich není mnoho. Vzhledem k tomu že je zdrojový kód programu překládán do interpretovaného MSIL, je zřejmé že jeho výkon nemůže dosáhnout ani lepších, ani stejných parametrů jako kód překládaný do nativního strojového kódu daného procesoru. Na druhou stranu poskytuje tomuto programu platforma .NET samozřejmě i své výhody. Díky nativní podpoře Unicode dokáže klient bez jakýchkoliv úprav nebo režií ze strany programátora zobrazovat všechny národní znakové sady, včetně arabských nebo asijských abeced. Také je velmi snadno rozšiřitelný o další funkce.

Samotný vývoj této aplikace pro autora představoval cenné programátorské zkušenosti. Programování se jako koníčku věnuje už několik let, projekt tohoto rozsahu ale prozatím nevytvářel, zvláště v .NET prostředí ne. Vyžadovalo to zvládnutí jiných pracovních návyků než při vytváření klasických malých aplikací. Zejména zde vynikla důležitost kompletního návrhu a důkladného promyšlení celkové struktury aplikace již před začátkem samotného vývoje. Bylo tedy nutné odolat pokušení přístupu „nejprve programovat a poté myslet“.

Při samotném programování nevznikly díky vynikajícím online dokumentacím žádné vážné problémy. Největší zádrhel představovala implementace UUEncode u dekódování binárních příloh, neboť popis systému UUEncode, který byl k dispozici, opomněl zdůraznit jedinou výjimku z pravidel pro převod bytů mezi použitými druhy kódování. Oprava byla nakonec otázkou jediného řádku kódu, ale jak už to při vývoji softwaru bývá, snaha o vyřešení tohoto problému zabrala celý den a vedla mnoha slepými uličkami. Únava z tohoto a podobných hledání správné cesty mezi jedničkami a nulami byla značná, radost z nalezených řešení však byla ještě větší.

Vytvořený program je plně funkční pro práci s Usenetem. Splňuje všechny standardy NNTP protokolu dané příslušnými dokumenty. Mezi desítkami jiných klientů, z nichž někteří jsou volně k dispozici na Internetu, není AlephNews svými vlastnostmi výjimečný,

nicméně podle dostupných zdrojů je to první NNTP klient vytvořený v prostředí .NET. Jeho vytváření přineslo autorovi kromě nových zkušeností také radost z tvůrčí práce.

SEZNAM POUŽITÉ LITERATURY

[1] BLUM Richard: *C# Network Programming*. Sybex 2003, ISBN: 0782141765

[2] KABELOVÁ, Alena, DOSTÁLEK, Libor. *Velký průvodce protokoly TCP/IP a systémem DNS*. 3. aktualizované vydání. Computer Press, a. s. 2002. ISBN: 80-7226-675-6

[3] ROBINSON, Simon, ALLEN, K.Scott, CORNES, Ollie. *C# Programujeme profesionálně*. Brno: Computer Press, a. s. 2003. 1130 s. ISBN: 80-7226-675-6

[4] SELLS, Chris. *C# a WinForms programování formulářů Windows*. Brno: Zoner Press, a. s. 2005. 648 s. ISBN: 80-86815-25-0

[5] TITUS, Robin: *C# Threading Handbook*. APress LLC 2004, ISBN: 1861008295

[6] *All about Usenet and Usenet history* [online]. 2004 [cit. 2006-06-06]. Dostupný z www: <http://www.smr-usenet.com/tech/about.shtml>

[7] *RFC 850 Standard for Interchange of USENET messages* [online]. 1983 [cit. 2006-05-19]. Dostupný z www: <http://rfc.net/rfc850.html>

[8] *RFC 977 Network News Transfer Protocol* [online]. 1986 [cit. 2006-05-21]. Dostupný z www: <http://rfc.net/rfc977.html>

[9] *RFC 1036 Standard for Interchange of USENET Messages* [online]. 1987 [cit. 2006-06-04]. Dostupný z www: <http://rfc.net/rfc1036.html>

[10] *RFC 2980 Common NNTP Extensions* [online]. 2000 [cit. 2006-05-21]. Dostupný z www: <http://rfc.net/rfc2980.html>

[11] *USENET Traffic Flow Maps* [online]. 2001 [cit. 2006-06-06]. Dostupný z www: <http://www.mundi.net/maps/maps_021/>

[12] *Uuencode - Wikipedia, the free encyclopedia* [online]. 2006 [cit. 2006-05-28]. Dostupný z www: <http://en.wikipedia.org/wiki/Uuencode>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

.NET	Moderní platforma firmy Microsoft pro vývoj aplikací
ARPANET	Advanced Research Projects Agency Network. Armádní síť, předek dnešního Internetu.
ASCII	American Standard Code for Information Interchange. Nejpoužívanější kódovací tabulka v informatice.
CRC	Cyclic Redundancy Check. Cyklický redundantní součet je speciální hašovací funkce používaná k detekci chyb během přenosu informací.
GUI	Graphical User Interface. Grafický mód ovládání programů.
INN	InterNetNews. Nejrozšířenější Usenetový server.
MIME	Multipurpose Internet Mail Exchange Extensions. Standard pro přenášení souborů pomocí emailu.
MSIL	Microsoft Intermediate Language. Mezikód do kterého jsou překládány .NET programy.
NNTP	Network News Transfer Protocol. Protokol, na němž funguje síť Usenet.
RFC	Request For Comments. Sada dokumentů popisující systémy Internetu.
TELNET	Telephone Network. Jeden ze základních síťových protokolů.
UML	Unified Modeling Language. Grafický jazyk určený pro modelování.
UNICODE	Průmyslový standard umožňující jednotné počítačové zobrazení všech národních znakových sad.
USENET	User´s network. Globální síť desetitisíců virtuálních bulletinů s diskusemi na všechna možná témata.
UUCP	Unix to Unix CoPy. Soubor dnes již nepoužívaných síťových protokolů.
UUENCODE	Unix to Unix Encoding. Algoritmu kódující binárních data do textu.
WINAPI	Windows Aplication Programming Interfaces. Sada C/C++ prostředků k programování operačního systému MS Windows.
YENC	Algoritmu pro kódování binárních dat do textových znaků.

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH