VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

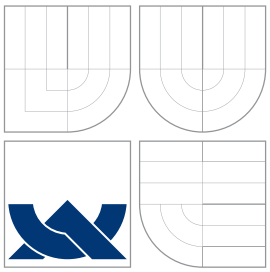
JABBER/XMPP TRANSPORT PRO DISKUZNÍ SKUPINY PŘÍSTUPNÉ PROTOKOLEM NNTP

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

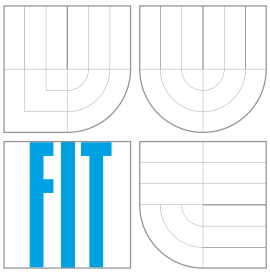
AUTOR PRÁCE
AUTHOR

PATRIK HALFAR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

JABBER/XMPP TRANSPORT PRO DISKUZNÍ SKUPINY PŘÍSTUPNÉ PROTOKOLEM NNTP

JABBER/XMPP TRANSPORT FOR NEWS VIA THE NNTP PROTOCOL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — PATRIK HALFAR
AUTHOR

VEDOUCÍ PRÁCE — Ing. MAREK RYCHLÝ
SUPERVISOR

BRNO 2008

Abstrakt

Tato práce se zabývá problematikou výměny informací v počítačových sítích. V první části se zabývá úvodem do diskuzních skupin, známých též pod názvem *NetNews* a shrnutím hlavních informací o nově se rozvíjejícím projektu *Jabber/XMPP*. V druhé části jsou popsány možnosti propojení těchto dvou protokolů; a detailněji popisuje aplikaci vytvořenou k tomuto účelu. V příloze je pak uvedeno, jak nakonfigurovat nejznámější servery pro spolupráci s touto službou. A pro úplnost jsou uvedena schémata požívaných XML souborů.

Klíčová slova

DNS, Instant messaging, NetNews, Mailing list, USENET, transport, klient, server, transport, služba, autentizace, autorizace, konečný automat,

Abstract

This document mention some possibilities of information sharing over computer networks. At the beginning there is described exchange information between group of users by *NetNews*. Next chapter make short introduction to project *Jabber/XMPP* and its possibilities. Other part appropriate to possibility combination these services and comment implemented application. There are include descriptions how configure most known Jabber servers for use of this application. There are contain *XML schemes* of uses files too.

Keywords

DNS, Instant messaging, NetNews, Mailing list, USENET, transport, client, server, transport, service, authentication, authorization, finite state machine,

Citace

Patrik Halfar: Jabber/XMPP transport pro diskuzní skupiny přístupné protokolem NNTP, bakalářská práce, Brno, FIT VUT v Brně, 2008

Jabber/XMPP transport pro diskuzní skupiny přístupné protokolem NNTP

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Markem Rychlým.

.....
Patrik Halfar
12. května 2008

Poděkování

Děkuji vedoucímu své bakalářské práce Ing. Marku Rychlému za jeho odborné vedení, vstřícnou spolupráci a podporu při psaní této bakalářské práce.

Obsah

Úvod

Dnešní doba je taková, že přístup k informacím není výsadou některých jedinců, nýbrž nutností se kterou se musí každý z nás denně potýkat. Možností jak informace rozšiřovat, vyměňovat, skladovat, aj. je v dnešní době bezpočet. Tato bakalářská práce se zabývá jen možnostmi, které se orientují na elektronické zpracování dat přes počítačové sítě.

Výměnu informací lze rozdělit podle několika kritérií. V rámci této práce jsou zajímavé hlavně tyto dvě – počet příjemců a čas mezi odesláním a přečtením. Při pohledu na hodnoty, jenž mohou tyto atributy nabývat, jsou zřejmé čtyři kombinace, které mohou nastat:

1. Jedinci okamžitě – Instant Messaging (*dále jen IM*)
2. Jedinci se zpožděním – E-mail
3. Skupině okamžitě – IRC
4. Skupině se zpožděním – NetNews, mailing lists

Po důkladnějším prozkoumání jednotlivých kombinací a jejích zástupců, si lze uvědomit, že každá má své jedinečné vlastnosti. Avšak u `mailing lists` by měl být brán v úvahu fakt, že nejde o konkrétní službu, nýbrž o rozšíření e-mailu.

Rovněž by mohla padnout otázka, proč zde nejsou uvedeny zástupci z kategorie *webových aplikací*. Na tuto otázku je odpověď jednoduchá – web jako takový nebyl pro tyto služby navržen, a musí složitými mechanismy řešit mnoho problémů, a nelze jej tedy považovat za „nativní službu“. Přesto v poslední době patří mezi ty nejrozšířenější a to zejména v oblastech, kde se jedná o komunikaci typu `many-to-many`.

I když se webové aplikace těší velkému úspěchu, přeci jen je neobvyklé využívání pro online služby. Pro tyto služby se nejvíce osvědčil IM, který je zastupován různými aplikacemi (ICQ, AOL Messenger, Yahoo Messenger, atd.). V poslední době se začíná rozšiřovat nový standard `Jabber/XMPP` (viz v kapitole 2).

Mezi nově se rozšiřující služby může být zařazeno i `RSS`. Zjednodušeně řečeno se jedná o službu, která nás upozorňuje na změny v offline službách – lze tedy říci, že z nich vytváří online služby. Problémem této služby je skutečnost, že je to pouze upozornění a je-li s daty nutno dále jakkoliv manipulovat, je třeba se k dané službě přihlásit. Mimo to, aby tato služba plnila svou funkci je třeba mít spuštěnou aplikaci, která bude kontrolovat RSS kanál.

Cílem této práce je nalezení efektnějšího způsobu, jak upozorňovat na změny v offline aplikacích přes IM a zároveň umožnit i manipulaci s těmito zdroji.

Současný stav

Při průzkumu současné situace s nabídkou aplikací orientovaných na propojení právě služby *NetNews* a IM na jádře Jabber/XMPP nebyl nalezen software, který by se této problematice věnoval. Řešení, jenž na trhu existují jsou orientovány na jiné protokoly, např. *SMTP – Simple Mail Transfer Protocol*, nebo byly řešeny pomocí robota. Tento způsob řešení představuje jediný kontakt, který se přidá do *rosteru* a odpovídá na zaslané příkazy. Výhodou tohoto řešení je fakt, že zprovoznění tohoto automatu nevyžaduje žádné nastavování na straně IM serveru. Naproti tomu za nevýhodu lze považovat, že pro všechny diskuzní skupiny existuje právě jeden kontakt, a to vyžaduje větší angažovanost od uživatele při ovládání. Řešení formou transportní služby umožňuje naopak pro každou diskuzní skupinu samostatný kontakt a v tomto důsledku je ovládání jednodušší. Za nevýhodu pak může být označena nutnost konfigurovat IM server pro spolupráci s touto službou. (viz v kapitole 3)

Kapitola 1

NetNews

1.1 Úvod

Počátky služby *NetNews* sahají až do devadesátých let. Konkrétněji lze mluvit o roce 1986, kdy byla vydaná první RFC specifikace společnosti IETF. Cílem bylo navrhnout službu pro sdílení informací mezi internetovou komunitou. Původně se hromadné informace šířily pomocí *mailing listu*.

1.2 Historie

NetNews nevznikl jen tak na čistém papíře, nýbrž byl vytvořen na základě již existujících protokolů. Konkrétněji lze mluvit o *RFC 822 – Standard for the format of ARPA Internet text messages*[1], později *RFC 2822* [17] a *RFC 850 – Standard for interchange of USENET messages*[5]. První z těchto protokolů, jak již název napovídá, definuje formát pro zprávy vyměňované na internetu (viz v kapitole 1.4). Druhému standardu se tato bakalářská práce nevěnuje, neboť byl nahrazen právě standardem *RFC 977 – Network News Transfer Protocol*[2], který v roce 2006 nahradil *RFC 3977* [3].

1.3 Princip

Základním principem *NetNews* je umístění všech diskuzních skupin na „jedno místo“. To neznamená, že existuje jediný server, ke kterému se přihlašují všichni uživatelé, ale serverů může být neomezeně a všechny zprávy (pokud nejsou nějak limitovány) se šíří na všechny servery. K této replikaci dochází v intervalech, které jsou nastaveny správcem. Výhoda, kterou to přináší uživateli spočívá jednoduše v tom, že mu stačí přihlásit se k odběru diskuzních skupin na jednom serveru.

Jak je zmíněno v předchozím odstavci, tak mohou existovat omezení, která určují, na jaký server se mohou dané skupiny replikovat. Dokonce lze toto omezení definovat i pro konkrétní zprávu. (viz v kapitole 1.6)

1.4 Formát zpráv

Tvar zpráv, jak již bylo uvedeno dříve, je definován standardem *RFC 2822 – Internet Message Format*[17].

1.4.1 Základní popis

Jednou z vlastností tohoto protokolu je, že pro zprávy používá textovou podobu. Ta je založená na sedmibitovém kódování `ASCII`, které kóduje znaky do číselných hodnot v rozsahu 1 až 127. Toto kódování známe jako `US-ASCII`. Je-li požadavek, aby zpráva obsahovala národní znaky, je nutné použít rozšíření MIME (*Multipurpose Internet Mail Extensions*) definované v *RFC 2045*, *RFC 2046*, *RFC 2047*, *RFC 2048* a *RFC 2049* [9, 10, 16, 7, 8]. Jednotlivé dokumenty popisují rozšíření pro různé části zprávy (hlavička, tělo) a nebudou v této bakalářské práci dále popisovány.

Zpráva vytvořena podle tohoto standardu je rozdělena do řádků. Každý z nich musí být ukončen dvojicí znaků *carriage-return* a *line-feed*. Jejich ASCII hodnoty jsou 13 a 10. Obvykle se pro tuto kombinaci používá značka `CRLF`.

Celá zpráva je rozdělena do několika částí: Nejprve jsou řádky s povinnými informacemi o zprávě, které mohou být doplněny o volitelné (rozšiřující) položky. Tento celek je nazýván „hlavička“, která je následována volitelnými daty – tělo zprávy. Hlavička a tělo jsou od sebe odděleny prázdným řádkem. Neboli prvním výskytem bezprostředně následujících páru `CRLF`.

1.4.2 Omezení délky řádku

Existují dvě omezení na délku řádku. První říká, že počet znaků na řádku nesmí přesáhnout hodnotu 998. A podle druhého by neměl přesáhnout 78 znaků. V obou případech bez dvojce `CRLF`.

První omezení vyplývá ze skutečnosti, že většina programů pro práci s internetovými zprávami (a to i dnes) si pro řádek alokuje buffer o velikosti 1000 znaků.

Druhý limit je doporučený a vyplývá z toho, že v textovém režimu se na jeden řádek vejde právě 80 znaků.

1.4.3 Hlavička

Popis hlavičky lze vyčíst v normě *RFC 1036 – Standard for Interchange of USENET Messages*[11], která definuje povinné i volitelné položky hlaviček vyměňovaných pomocí protokolu NNTP (viz v kapitole 1.5). Je nutno podotknout, že norma připouští i neznámé hlavičky, a ty je nutné přeposílat.

Mezi povinné položky patří „From“, „Date“, „Newsgroups“, „Subject“, „Message-ID“ a „Path“. Jako volitelné položky lze uvést „Followup-To“, „Expires“, „Reply-To“, „Sender“, „References“, „Control“, „Distribution“, „Keywords“, „Summary“, „Approved“, „Lines“, „Xref“ a „Organization“. Všechny tyto položky jsou vysvětleny níže.

1.5 Výměna zpráv

Již v úvodu je zmíněno, že výměna zpráv je zajištěna protokolem NNTP definovaném v *RFC 3977 – Network News Transfer Protocol*[3]. Stejně jako formát ukládání zpráv tak i komunikace je textová. Její princip je velice jednoduchý a spočívá na systému dotaz – odpověď. Za dotaz se považuje klíčové slovo, které je v některých případech doplněno argumentem. Odpověď je strukturovanější. Základem je třímístný číselný kód, ze kterého lze vyčíst (ne)úspěšnost operace a případně i důvody neúspěchu, následován textovým popisem. V některých případech je tento řádek následován textem, který tvoří odpověď a zpravidla je ukončen řádkem obsahujícím jen tečku (`.`).

1.5.1 Vybrané příkazy

Zde je uvedeno několik základních příkazů NNTP protokolu, které jsou použity v aplikaci.

- LIST
- GROUP
- NEXT, LAST
- HEAD, BODY
- POST

LIST Tento příkaz umožňuje vypsat seznam všech diskuzních skupin dostupných na serveru. Odpovědí na tento příkaz je zpráva s kódem `215` následována výčtem, který je ukončen tečkou na samostatném řádku.

GROUP Aby bylo možné čísti příspěvky je nutno být přepnut právě do té diskuzní skupiny, o jejíž příspěvky se jedná. Tuto funkci plní právě tento příkaz, jehož argumentem je název diskuzní skupiny, do které se přepíná. Na tento příkaz existují dvě možné odpovědi:

1. `211` – úspěch; tento kód je následován trojici čísel (počet příspěvků, číslo nejnižšího a číslo nejvyššího) a názvem skupiny
2. `411` – neúspěch; znamená, že diskuzní skupina nebyla nalezena

NEXT, LAST Příkazy slouží k přepínání na následující, resp. předchozí příspěvek. Jsou bez argumentu, a umožňují se pohybovat pouze o jeden příspěvek. Na oba příkazy existují čtyři varianty odpovědí: jedna kladná (`223`) a tři negativní (`412` – Není zvolena diskuzní skupina; `420` – Diskuzní skupina je prázdná; `421`, resp. `422` – Již je vybrán poslední, resp. první příspěvek v diskuzní skupině).

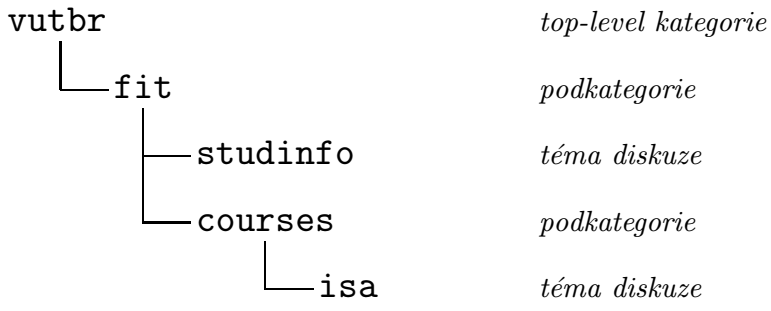
HEAD, BODY Tyto dva příkazy slouží ke čtení vybraného příspěvku. Další možností je tyto příkazy doplnit argumentem (*message-id* nebo *číslo příspěvku*). `HEAD` zobrazí jen hlavičku zprávy a `BODY` zas jen její tělo. Oba příkazy lze nahradit příkazem `ARTICLE`, který obojí vypíše najednou. Proč tento příkaz není použit v této bakalářské práci je vysvětleno v kapitole 3.2.2.

Z principu věci jsou tyto příkazy použity bez argumentů. V této situaci může odpověď nabývat třech stavů. Vykonáno úspěšně (`221`) – za touto odpovědí následuje hlavička/tělo zprávy. Není vybrána diskuzní skupina (`412`). Neexistuje příspěvek s tímto číslem (`420` u `HEAD`; `423` pro `BODY`).

POST Pomocí tohoto příkazu se posílá příspěvek na server. Komunikace probíhá ve dvou fázích. Nejprve je zasláno řídící slovo `POST` bez argumentů. Je-li reakci na tento příkaz kód `440`, pak není povoleno přidávat příspěvky. V opačném případě je odpovědí `340`. V této situaci server očekává, že mu klient zašle zprávu ve formátu uvedeném v kapitole 1.4. Zpráva je zpracována, po zaslání tečky na samostatném řádku. Kódem `240` je potvrzeno úspěšné nahrání a `441` označuje selhání.

1.6 Organizace diskuzních skupin

vutbr.fit.studinfo
vutbr.fit.courses.isa



Obrázek 1.1: Členění diskuzních skupin

Diskuzní skupiny na *NetNews* serveru jsou členěny hierarchicky[25]. Existuje několik tzv. *top-level* kategorií, které mají globální charakter, šíří se tedy mezi všemi servery bez omezení. Naproti tomu mohou být na některých serverech „lokální“ kategorie, které se vůbec nešíří na další servery, nebo jen na některé. Příkladem je *top-level* kategorie `vutbr` dostupná na serveru `nntp://news.fit.vutbr.cz/`.

Tyto *top-level* kategorie se mohou dále členit na další podkategorie, nebo mohou obsahovat již konkrétní téma. Jako oddělovací znak se používá stejně jako v *URL* tečka (`.`), avšak na rozdíl od *URL* je nejvyšší úroveň vlevo. Název diskuzní skupiny je tvořen názvem tématu, včetně jeho nadřazených kategorií (viz Obrázek 1.1).

1.6.1 *Top-level* kategorie

Pro úplnost jsou zde uvedeny globálně šířené kategorie a jejich význam.

alt	značí *Alternativní kategorie*. Zde se obvykle nacházejí témata, která jsou kontroverzní (např. `alt.sex`). A ostatní se zde nacházejí protože jejích vytvoření v této kategorii je snadné
comp	shromažďuje témata zaměřená na počítače a věci okolo nich
humanities	je kategorie zaměřená na humanitní témata, jako jsou filozofie, umění, literatura, apod
misc	je kategorie, do které se umísťují příspěvky, které nelze zařadit jinam
news	obsahuje příspěvky a podkategorie věnující se *NetNews* samotnému
rec	se zaobírá tématy věnovaným koníčkům a zálibám
sci	zastřešuje vědecky orientovaná témata
soc	se zaměřuje na sociální problémy
talk	slouží k rozsáhlejším diskuzím

Tabulka 1.1: Přehled *Top-level* kategorií

Kapitola 2

Jabber/XMPP

Jak již bylo uvedeno, tak služba Jabber/XMPP je jedním ze zástupců tzv. *Instant Messaging* služeb, což jsou služby pro online komunikaci mezi uživateli. Základem celé komunikace je *XMPP – Extensible Messaging and Presence Protocol*, jehož základní vlastností je otevřenost. Nad tímto protokolem vznikla „platforma“ a ta je nazývaná *Jabber*. (viz v kapitole 2.2).

Základní výhodou této platformy oproti ostatním zástupcům IM je její otevřenost a decentralizovanost. Otevřeností lze rozumět jak to, že vychází z uveřejněných standardů, tak i to, že nevyžaduje používání konkrétního klienta, který např. obtěžuje uživatele reklamou. Decentralizovanost zas umožňuje provozovat vlastní server třeba na lokální síti a ten může dokonce komunikovat bez problému s ostatními servery na internetu.

Dalších výhod je mnoho, a snad všechny pramení právě z otevřenosti protokolů, jako příklad lze uvést možnost libovolně tento protokol rozšířit jak o vlastní funkce, tak i o služby, které může server nabízet. Názorným příkladem je *Transport mezi protokoly XMPP a NNTP*, vytvořen v rámci této bakalářské práce.

2.1 Historie

První zmínka o otevřeném protokolu pro komunikaci v reálném čase se datuje do roku 1999. Kdy vznikl první server, a pár klientů podporující komunikaci ve formátu *XML* [4].

V roce 2000 uveřejňuje IETF první výsledky pracovní skupiny IMPP (*Instant Messaging and Presence Protocol*), kterými jsou dva standardy. První z nich popisuje abstraktní model pro prezenční a IM systémy (*RFC 2778 – A Model for Presence and Instant Messaging*[13]). Druhým standardem je *RFC 2779 – Instant Messaging / Presence Protocol Requirements*[14], který popisuje požadavky na tyto systémy. V červnu téhož roku vydává první návrh dokumentující základ protokolu *Jabber*. Avšak pro špatnou organizaci komunity platnost návrhu vypršela dříve než byl schválen.

Rok po té vzniká *Jabber Software Foundation (JSF)*, která zastřešuje celý projekt a ještě v témže roce vydává rozšiřující standard *XEP-0001* (viz v kapitole 2.4), který

popisuje, jak vytvářet další rozšíření.

Na přelomu let 2001 a 2002 se opět začalo pracovat na základním protokolu, ale tentokrát pod záštitou *JSF* a opět je podán jako návrh standardu organizaci IETF. V reakci na úspěch tohoto návrhu vznikne pracovní skupina IETF pro formalizaci základního protokolu *Jabber*, která dodnes vystupuje pod názvem *Extensible Messaging and Presence Protocol (XMPP)*. Již ke konci roku byly podány první návrhy pro standardizaci („XMPP Core“ a „XMPP IM“).

V roce 2003 vznikají a schvalují se dokumenty, na kterých bude tato technologie stavět. Mezi tyto dokumenty lze zařadit například *RFC 4422 – Simple Authentication and Security Layer (SASL)*[12] dříve znám jako *XEP – 0034*, *RFC 4346 – The Transport Layer Security (TLS) Protocol*[6] a další.

Rok 2004 je zlomovým, neboť dochází ke zveřejnění klíčových standardů.

- *RFC 3920 – Extensible Messaging and Presence Protocol (XMPP): Core*[18]
- *RFC 3921 – Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence*[19]
- *RFC 3922 – Mapping the Extensible Messaging and Presence Protocol (XMPP) to Common Presence and Instant Messaging (CPIM)*[21]
- *RFC 3923 – End-to-End Signing and Object Encryption for the Extensible Messaging and Presence Protocol (XMPP)*[20]

Následujícího roku se do vývoje zapojuje společnost *Google* a vyvíjí vlastní službu postavenou na této platformě – *Google Talk*.

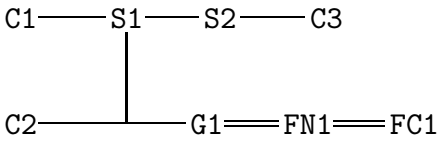
V dalších letech vznikne jen několik standardů RFC. *RFC 4622 – Internationalized Resource Identifiers (IRIs) and Uniform Resource Identifiers (URIs) for the Extensible Messaging and Presence Protocol (XMPP)*[22] v roce 2006, který je v únoru roku 2008 nahrazen dokumentem *RFC 5122*[24] nesoucím stejný název. V roce 2007 se objevují dva poslední RFC dokumenty *RFC 4854 – A Uniform Resource Name (URN) Namespace for Extensions to the Extensible Messaging and Presence Protocol (XMPP)*[23] a *RFC 4979 – IANA Registration for Enumservice 'XMPP'*[15]. Další standardizační dokumenty se vydávají pod označením *XMPP Extension Protocol (XEP)*.

2.2 Princip

V úvodu této kapitoly je naznačeno, že celá komunikace je vystavěna na formátu *XML* a základ architektury je tvořen pouze základními vlastnostmi tvořící jádro platformy. Tyto vlastnosti umožňují pouze přidání/odebrání uživatele, informace o prezenci a zasílaní zpráv. Koncepce je tedy rozdělena na dvě části: jádro systému a rozšíření.

2.3 Jádro XMPP

Mimo vlastností uvedených výše, je třeba řešit způsob vytvoření spojení a to nejen mezi klientem a serverem, ale i mezi servery (tato potřeba vyplývá z vlastnosti decentralizace). Lépe je tato situace znázorněna na obrázku 2.1. Z něhož je zřejmé, že budou-li spolu chtít komunikovat klienti `C2` a `C3`, tak spolu musí navázat spojení jejich servery (`S1` a `S2`). Stejná situace nastává i v případě, když klient `C1` bude chtít komunikovat s klientem cizí sítě `FC1`. Touto komunikací se podrobněji zabývá kapitola 3.



Obrázek 2.1: Architektura komunikace

Legenda k obrázku:

- Jednoduchá vazba (—) označuje komunikaci protokolem XMPP.
- Dvojitá vazba (=) označuje komunikaci jiným protokolem (v rámci této bakalářské práce si lze doplnit protokol NNTP).
- **C1, C2, C3** jsou klienti podporující protokol XMPP.
- **S1, S2** jsou servery pro službu *Jabber*.
- **G1** představuje bránu, která zprostředkovává komunikaci mezi jiným typem sítě a protokolem XMPP.
- **FN1** jiná síť (např. *NetNews*).
- **FC1** klient podporující protokol jiné sítě.

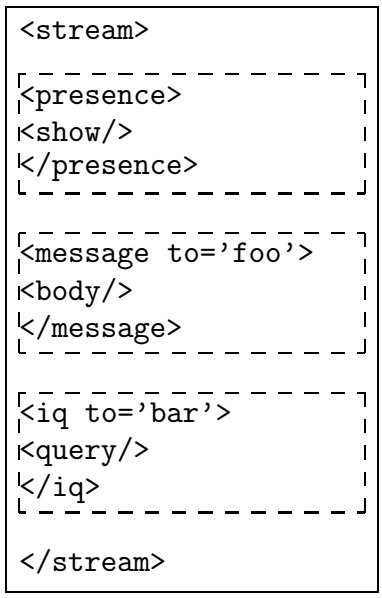
2.3.1 Způsob adresování

Mezi mnoha uživateli je jako zástupce IM známo *ICQ*. Tato služba používá pro adresování klientů číslo. Tento způsob lze označit za nevýhodný především z toho důvodu, že čísla se hůře pamatují a nemohou nést vedlejší informace o uživateli. *Jabber* nepřichází s žádným novým způsobem pro adresování klientů, nýbrž přebírá stejné schéma jenž se používá u e-mailových adres (*jméno@doména*). Výhodou tohoto schématu je praktičtější uplatnění např. ve firemním prostředí, protože z takovéto adresy již lze vyčíst z jaké firmy je adresát. Mimochodem stojí za zmínění, že licenční smlouva pro používání *ICQ* vylučuje nasazení v komerční sféře.

Jabber je v současné době asi jedinou službou pro IM, která podporuje připojení z několika míst současně. S přihlédnutím k této skutečnosti, nestačí mít jednoznačnou identifikaci uživatele, kterou nám zajišťuje adresa výše uvedeného formátu. A proto k tomu byla zavedena identifikace zdroje (místa odkud je uživatel připojen): úplná adresa má tedy tvar: *jméno@doména/zdroj*.

2.3.2 Průběh spojení („sezení“)

„Sezení“ je termín, kterým se označuje celý průběh spojení (od připojení k serveru a autentizace, přes posílaní zpráv, až po ukončení spojení). Celý tento průběh lze vidět na obrázku 2.2. Části uzavřené v přerušovaném obdélníku se mohou vyskytovat opakovaně v libovolném pořadí.



Obrázek 2.2: Průběh „sezení“

Sezení zahajuje klient, který ihned po připojení zasílá serveru tág `<stream>`. Ten odpovídá zasláním stejné zprávy. Případně může zaslat element `features`, ve kterém klienta informuje o možnostech autentizace a klient se autentizuje, je-li úspěšný, zasílá se ještě jeden element `<stream>`, který již otevírá prostor pro komunikaci. (viz RFC 3920 [18])

Po té, co je otevřen stream pro komunikaci, proběhne ještě nahrání *rosteru* (seznamu kontaktů) ze serveru na klienta. (Ukládání seznamu kontaktů na serveru umožňuje současné připojení z několika míst, při zachování plné kontroly nad kontakty.) Nyní již klient může komunikovat s ostatními uživateli, tzn. zasílat jim zprávy, informace o svém stavu, či požadavky typu `iq` (Info/Query).

Z obrázku 2.2 se může zdát, že protokol XMPP podporuje jen málo příkazů a to by mohlo být omezující. Opak je však pravdou. Jak je známo, tak XML elementy nemají jasnou sémantiku, a tak je tomu částečně i v případě XMPP. Některé příkazy mají sémantiku implicitní, a jiným je dodána přiřazením do tzv. *jmenného prostoru*. Výhoda tohoto řešení spočívá v tom, že při použití vlastnosti, která má význam ryze na klientech, není nutná žádná podpora ze strany serveru, neboť značku zvládne zpracovat vždy, aniž by se díval na její sémantiku. (Seznam jmenných prostorů lze nalézt na stránce *http://www.xmpp.org/schemas/*, kde je v tabulce vidět, kterým rozšiřujícím protokolem je prostor specifikován.)

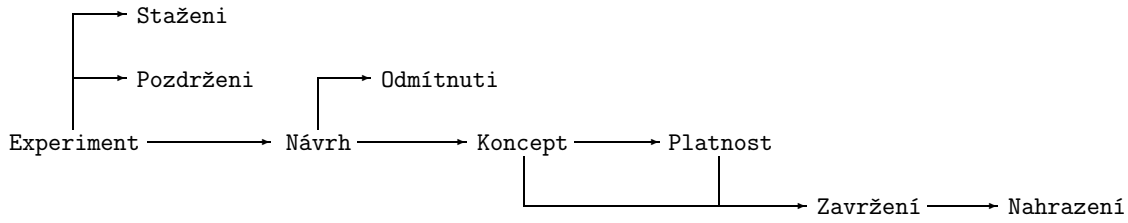
Spojení může ukončit jak klient, tak i server zasláním značky `</stream>`. Na tuto zprávu musí druhá strana odpovědět zasláním stejné zprávy a ukončením spojení.

2.4 Rozšíření XMPP

Všechny další možnosti, jako je vytváření „chatovacích místností“, transportní služby, zasílaní souboru jsou definovány jako rozšíření. Ta jsou dostupná v dokumentech označených jako *XEP*. Některé z těchto rozšíření jsou pro služby serveru, jiné rozšiřují pouze vlastnosti klientů, bez nutnosti podpory ze strany serveru (např. formuláře). Chce-li kdokoliv tato rozšíření využívat, je třeba, aby je podporovala i druhá strana. Dále jsou popsána ta rozšíření, na která je odkazováno v kapitole 3.

2.4.1 XEP-0001 — XMPP Extension Protocol

Tento protokol vznikl mezi prvními a jeho význam je zásadní, neboť definuje, pravidla, pro vytváření dalších rozšíření. Mimo jiné definuje i „životní cyklus“ rozšiřujících protokolů. (Viz Obrázek 2.3)



Obrázek 2.3: „Životní cyklus“ protokolu

2.4.2 XEP-0004 — Data Form

Rozšíření XEP-0004 je především rozšíření klienta. Zavádí do komunikace formulářové prvky, jako jsou tlačítka, rozbalovací seznamy, apod. Data nese element `x`, který je doplněn o sémantiku jmenným prostorem. Při jeho posílání musí být zapouzdřen buďto v elementu `message` nebo `iq`.

2.4.3 XEP-0030 — Service Discovery

V tomto případě se jedná především o rozšíření vlastností serveru. Service Discovery umožňuje prohlížet ve stromové struktuře služby nabízené serverem. Zavádí dva jmenné prostory: *http://jabber.org/protocol/disco#info*, který slouží k výměně informací o službě a *http://jabber.org/protocol/disco#items* pro výměnu poddružených položek (tím vzniká stromová struktura).

2.4.4 XEP-0054 — vcard-temp

VCard tvoří vizitku, kterou si mohou nastavit nejen uživatelé, ale může jí mít nastavenou i služba. Jak již samotné přirovnání k vizitce napovídá, obsahuje obvykle

osobní informace o uživateli, které mohou být umístěny do polí s konkrétním významem, nebo které mohou být uvedeny jako poznámka. U služby může položka pro poznámky obsahovat informace o změnách, aj.

2.4.5 XEP-0077 — In-Band Registration

Obecně je pravidlem, že server nemusí nabízet všechny služby automaticky všem uživatelům. Většina severů nabízí zdarma registraci, používání účtu a některé základní služby. K tomuto základu pak můžou ještě nabízet „VIP“ služby – ty jsou obvykle placené a proto jsou dostupné jen některým uživatelům. Díky možnosti registrace služeb, lze tuto situaci řešit velice elegantním způsobem, kdy k odběru služby jednoduše můžeme požadovat autentizaci.

Registrace nemusí vždy sloužit k tomu, aby autentizovala uživatele. Její využití může také spočívat v parametrizaci služby. Kdy si uživatel, který si službu registruje, vybírá její možností.

2.4.6 XEP-0092 — Software Version

Zde se jedná o velice jednoduché rozšíření, které dovoluje klientovi zeptat se, jakou verzi má vzdálená strana. Mimo to může nést i informaci o operačním systému druhé strany.

2.4.7 XEP-0100 — Gateway Interaction

V případě tohoto rozšíření lze hovořit o tzv. informačním protokolu. Jeho smysl spočívá v tom, že definuje interakci mezi XMPP a jinými protokoly. Zároveň doporučuje některé způsoby řešení.

2.4.8 XEP-0114 — Jabber Component Protocol

Protokol popisuje způsob, jak se k serveru připojuje služba. Definuje jmenný prostor, pro element `stream`. Dále definuje způsob autorizace. Ten se provádí pomocí zadání *handshake*, který se vypočítává ze sdíleného klíče a náhodné hodnoty jako MD-5 hash.

2.4.9 XEP-0202 — Entity Time

Protože *Jabber* může nabízet i jiné služby, než prostou komunikaci, je vhodné mít způsob, jak zajistit, že obě strany mají přehled o tom, jaký čas má strana druhá.

Nejnázorněji to lze uvést na příkladě: Letecká společnost spustí možnost rezervace letenek přes *Jabber* robota a uživatel si bude chtít rezervovat letenku. Avšak robot nebude vědět, že uživatel má jinak nastavené časové pásmo, a tak letenku zarezervuje na jiný čas. Kdyby obě strany podporovaly výměnu časové informace, robot by mohl uživatele informovat, že má jiný čas než on a uživatel by tuto informaci mohl vzít v potaz.

Kapitola 3

Vlastní řešení – Transport mezi službami

Tato kapitola bakalářské práce se podrobněji zabývá transportní službou mezi protokoly XMPP a NNTP. Detailněji jsou zde popsány některé konstrukce a důvod jejích požití. V této části jsou také navrženy i další možnosti rozšíření, které nejsou implementovány, neboť cílem bylo vytvořit základ, který by byl stabilní, nikoliv robustní.

V několika případech bylo nutné navrhnout i vlastní soubory (např. konfigurační). Aby byla dodržena jednotnost, mají tyto soubory formát XML dokumentu. Jejích schéma je přílohou této bakalářské práce.

3.1 Úvod do problematiky

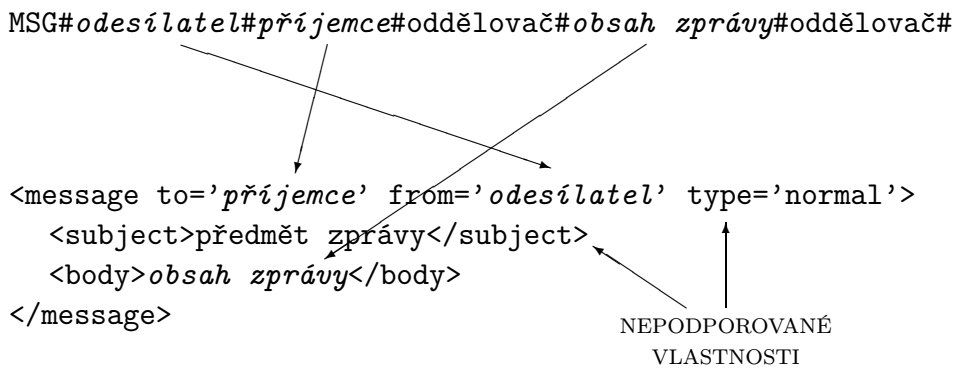
Smyslem transportu je propojit dva různé protokoly. Již existují služby, které propojují XMPP s ICQ, nebo s MSN Messenger, a další IM služby. Propojení služeb stejného charakteru (tedy IM) není tak komplikované (za předpokladu, že jsou oba dva protokoly známé). Celý princip spočívá v nahrazení hlavičky jednoho protokolu za hlavičku protokolu druhého (viz Obrázek 3.1). Samozřejmě to není to jediné co se musí řešit, velkým problémem je adresování. To lze v XMPP řešit velmi elegantně, při využívání transportu (viz v kapitole 3.1.2). Komplikace nastávají, nabízí-li jeden protokol nějaké rozšiřující možnosti, které druhý protokol nepodporuje – pak se tyto vlastnosti odstraňují.

Cílem této bakalářské práce bylo vytvořit transport mezi dvěmi službami, které jsou každá jiného charakteru, tedy IM, zastoupen službou *Jabber*, a diskuzní službou *NetNews*. Rozdíl těchto dvou služeb je zřejmý již při prvním pohledu:

- *Jabber* – doručuje zprávy adresátovi v reálném čase, je-li online,
- *NetNews* – zprávy jsou doručovány do schránky na serveru, kde si je uživatelé mohou přečíst.

Je tedy jasné, že problém se musí řešit trošku komplikovaněji, než je tomu u transportu mezi službami stejného charakteru. Zde je nutné řešit přístup ke zprávám. Ty musí být uživateli dostupné na vyžádání, tak jak je tomu u standardního přístupu

přes nativního klienta služby *NetNews*. Aby toto rozšíření zvýšilo uživateli možnosti, zavádí se zde i událost „trap“, která vznikne při nahrání nového příspěvku na server.



Obrázek 3.1: Transport IM služeb

Podrobnější popis činnosti *NetNews* serveru je v kapitole 1. Zde je jen připomenuto, že se obecně diskuzní skupiny a jejích zprávy šíří i na další servery. K této synchronizaci dochází v časových intervalech, a tak i událost „trap“ se generuje při pravidelné kontrole zpráv na severu.

Navrhovaná služba musí řešit problém přístupu ke zprávám na server. Možnosti jak tento problém řešit jsou tři:

1. Pokaždé se připojit k serveru a vyhledat konkrétní zprávy
2. Uložit si hlavičky zpráv, a pro jejich obsah se připojit k serveru
3. Vytvořit si lokální kopii, která se v pravidelných intervalech synchronizuje

Každá z jmenovaných vlastností má jistě svá pro i proti. A řešení spočívá na volbě vhodného kompromisu. Jako řešení byla nakonec vybrána poslední možnost. Její nevýhodou je vytváření lokální kopie zpráv, ale s přihlédnutím na velikosti zpráv a dnešních kapacit disků je tento problém zanedbatelný, a výhoda, kterou toto řešení přináší je rychlost a dostupnost.

Nyní, když už jsou vyřešeny hlavní problémy, se kterými je třeba počítat, lze přistoupit k analýze možného způsobu implementace. Tedy konkrétněji, zda službu implementovat jako robota, nebo transport.

3.1.1 Robot – kontakt

Tento způsob implementace je velice jednoduchý, ve své podstatě se jedná o konečný automat, který vystupuje jako jeden kontakt, a celé řízení je závislé na příkazech, které se tomuto automatu zašlou. Řízení je stejné jako u služby, rozdíl spočívá spíše v tom, že u robota musí být každý příkaz parametrizován, aby bylo zřejmé s čím si uživatel přeje pracovat (v tomto konkrétním případě lze říci, že v každém příkazu

musí být uvedeno, o kterou diskuzní skupinu se jedná). V kontrastu s tímto je jednodušší implementace, neboť i pro skriptovací jazyky dnes existují knihovny, které lze efektivně využít.

V úvodu této bakalářské práce je zmíněno, že tento způsob nevyžaduje žádný zásah na straně serveru. Jednoduše se na serveru zaregistruje libovolný kontakt a tomuto robotovi se nastaví, aby se k serveru přihlašoval pod tímto účtem a heslem, které mu náleží. Pokud tedy není možné zasáhnout do nastavení serveru, je to jediné možné řešení.

3.1.2 Transport – služba

Implementace služby již není tak jednoduchá jako samotný robot, především musí zahrnovat již v základu některé rozšiřující protokoly (viz v kapitole 2.4), a také samotné schéma je poněkud složitější. Ať už proto, že se nekontroluje jestli je zpráva určena jen službě, ale i to, zda je určena některé z podřízených součástí.

Tímto byla zmíněna základní výhoda tohoto řešení. Služba se neadresuje jako kontakt, nýbrž jen jako server (doména). Z toho vyplývá, že jako kontakt může být (v tomto konkrétním případě) označena každá diskuzní skupina samostatně. Toto řešení samozřejmě způsobí více kontaktu v *rosteru*, na druhou stranu má uživatel přehled, které diskuzní skupiny odebírá, a chce-li s některou pracovat, nemusí používat parametr, aby serveru sdělil, se kterou diskuzní skupinou chce zvolenou činnost provádět.

Aby bylo možné toto řešení využít je nutné zpřístupnit tuto službu na serveru. Protože v této bakalářské práci byla vybrána právě tato varianta, je v příloze A uvedena konfigurace nejpoužívanějších *Jabber* serveru.

3.2 Zvolené řešení

Z předchozí kapitoly vyplývá, že implementovaná služba je navržena jako transport, který si vytváří lokální kopie zpráv.

Před detailnějším popisem implementované aplikace, je uvedena vsuvka o zvoleném jazyce, v němž je aplikace napsána. Je nutné si uvědomit, že každý uživatel může pracovat v jiném prostředí. Tomu odpovídá i dostupnost klientů pro různé platformy. I přesto, že cílem této bakalářské práce bylo navrhnout a vytvořit transportní službu, u které by nebylo nutné řešit nezávislost na operačním systému, byla platformová nezávislost jedním z kritérií. Po analýze vlastností různých programovacích jazyků byl zvolen jako implementační jazyk *Java*. Programy napsány v tomto jazyce běží na virtuálním stroji, a nezávisí tedy na operačním systému. Mimo jiné je v tomto prostředí velice dobrá podpora vláken a tedy paralelizace. Mezi další výhody lze zařadit objektovou orientaci tohoto jazyka.

3.2.1 Základní koncept

Jádro aplikace je tvořeno ze dvou na sobě nezávislých vláken. Jedno pro komunikaci s XMPP serverem a to druhé pro službu *NetNews*. Po spuštění aplikace se spouští

obě dvě vlákna, první se připojí k *jabber* serveru, a toto spojení je permanentní. Vlákno pro službu *NetNews* se připojí k NNTP serveru, a zkontroluje jestli jsou na serveru nové příspěvky. Po té co dokončí tuto činnost se uspí. Tuto činnost opakuje v pravidelných intervalech. Konkrétní činnosti jednotlivých vláken jsou popsány níže.

3.2.2 NetNews

Nejprve je popsáno to jednodušší vlákno. Jeho princip je velice jednoduchý. Připojí se k serveru, vyžádá si seznam všech diskuzních skupin, ty které jsou povoleny uloží a posléze si pro všechny uložené skupiny vyžádá seznam zpráv. Pokud již má nějaké skupiny uloženy, pak předtím, než začne zjišťovat nové zprávy, všechny ve skupině označí příznakem `deleted`. Kontrola zpráv probíhá tak, že nejprve stáhne hlavičku zprávy a zjistí, zda již má tuto zprávu uloženou. Pokud ano, zruší její příznak `deleted`. Jestliže tuto zprávu nemá dojde k vyžádání jejího těla a zprávu uloží. Této zprávě je přidělen příznak `new` a zároveň nastaví privátní signifikant, že byla přidána nová zpráva. Následně přejde na další zprávu, promocí příkazu `NEXT`. Tyto kroky se provádějí iterativně, dokud odpovědi na příkaz `NEXT` není zpráva s kódem `421` (viz v kapitole 1.5.1).

Ačkoli je pro kontrolu, zda je zpráva již uložena, nutné znát jen *Message-ID*, které je obsaženo v odpovědi na příkaz `NEXT`, není této možnosti využito. Naopak je využit postup popsaný výše, kdy se stahuje celá hlavička, ve které se *Message-ID* vyhledává. Důvodem tohoto řešení je požadavek na uniformitu zpracování. Tu nelze zařídit nastíněnou možností, kde by se využíval příkaz `NEXT` a stažení zprávy by zajišťoval příkaz `ARTICLE`, neboť první zpráva by musela být zpracována odlišným způsobem než ostatní. Pro oddělené zpracování hlavičky a těla zprávy, bylo přistoupeno i z toho důvodů, že zprávy mohou obsahovat přílohy, které jsou mnohonásobně větší, než samotná zpráva. Ve zvoleném řešení dochází tedy ke stažení přílohy jen jedenkrát.

Po té, co jsou takto zpracovány všechny povolené diskuzní skupiny, je provedená kontrola, zda nejsou uloženy příspěvky, které mají být nahrány na *NetNews* server. Po jejích zpracování se spojení ukončí a byl-li nastaven lokální signifikant nové zprávy, překlopí se na globální.

Ukládání zpráv

Stažené zprávy jsou ukládané do souboru XML (XML schéma je v příloze B.2), který se efektivně zpracovává pomocí knihovny *JDOM*. O zprávě nejsou ukládány všechny informace, které o ní má k dispozici *NetNews* server, ale jen některé položky („From“, „Date“, „Subject“, „Message-ID“, „References“). V současné době položka references není nějak zpracovávána. Je to především díky neúplné podpoře funkce `thread` XMPP protokolu ze strany klientů. Alternativní přístup zatím nebyl řešen.

Další ukládanou informací o zprávě je její tělo. V současné době aplikace dokáže zpracovat i vícenásobný obsah zprávy, který je vytvořen pomocí standardu *MIME*. Pokud zpráva obsahuje přílohy, vždy si ukládá jejích názvy a datové typy. Následné zpracování přílohy závisí na nastavení serveru. (viz v kapitole 3.2.4)

Připojování k serveru

Obecně z popisu činnosti *NetNews* serveru vyplývá, že se stačí připojit k jedinému serveru, pro možnost čtení všech diskuzí. Problém vzniká, když jsou na serveru použity nějaké omezení, která zabrání replikaci zpráv na jiný server. Z tohoto důvodu je tento *NetNews* klient navržen tak, že se může připojovat k několika serverům. Ke kterým se má připojit se nastavuje v konfiguračním souboru. Dalšími parametry, kterými lze připojování ovlivnit jsou: počet pokusů o připojení, časová prodleva mezi nimi, interval kontroly nových zpráv. (viz v kapitole 3.2.4)

3.2.3 Jabber

Druhým vláknem v programu je komunikace s *Jabber* serverem. To je poněkud složitější, neboť služba, po té co se autorizuje a naváže spojení se serverem, přechází do režimu naslouchání a zpracovávání zpráv, či příkazů od uživatelů.

Připojení k serveru

Jak probíhá „sezení“ se serverem je principiálně popsáno v kapitole 2.3.2. Zde je věnována pozornost rozdílům mezi připojením klienta a připojením služby. Zahájení probíhá zasláním stejného tágu `stream` jako v případě klienta. Rozdíl je v tom, že tento tág patří do jiného jmenného prostoru. Dále proběhne autorizace. Pokud je úspěšná, je již dále služba připravena naslouchat. Průběh této komunikace je popsán v protokolu *XEP-0114*. s Následná komunikace se serverem probíhá již standardně jako komunikace mezi dvěma klienty, tedy zasílají se prosté zprávy, či požadavky *Info/Query*. Detailněji je další činnost popsána v kapitole 3.3 Ovládání.

I když komunikace probíhá pomocí zasílaní textu ve formě XML, není zpracovávána pomocí knihovních funkcí, neboť nemá plnou strukturu XML dokumentu, dokud nedojde k uzavření komunikace. Z těchto důvodů bylo přistoupeno k parsování této komunikace pomocí běžných funkcí pro práci s textem.

3.2.4 Konfigurace

Všechny základní vlastnosti služby se konfigurují v jediném konfiguračním souboru, ten je ve formátu XML (XML schéma je v příloze B.1). I tento soubor je zpracováván pomocí knihovny *JDOM*.

Soubor je rozdělen do čtyř částí: XMPP, NNTP, lokální konfigurace, jazyková konfigurace. Dále není rozebírána struktura dokumentu, ale následuje popis sémantického významu klíčových prvků.

XMPP

`jid`	Jabber ID, pod nímž tato služba vystupuje. Hodnota musí být opět stejně nastavena i v konfiguraci serveru. (Označení `__JID__` v příloze A)
`server`	Server, ke kterému se připojuje. Ten může být uveden pomoci IP adresy, nebo DNS jména.
`shared_key`	Zde se zapisuje fráze, která je uvedena i na straně serveru. Aby mohlo dojít k úspěšné autorizaci, musí byt obě fráze stejné.
`timeout`	Hodnota představuje čas v milisekundách. Pokud během této doby nedojde k připojení, je server označen za nedostupný.
`reconnect`	Obsahem je čas v sekundách, který tvoří prodlevu mezi opětovnými pokusy o připojení.
`show_item`	Je-li tento tág přítomen, jsou diskuzní skupiny zobrazeny všem. V opačném případě se zobrazují jen registrovaným uživatelům.
`vCard`	Tento prvek má stejný sémantický význam jako vCard popsán v dokumentu *XEP-0054*, včetně jeho synovských prvků. Které prvky jsou přípustné je zřejmé ze schématu v příloze B.1.

NNTP

`server`	Server na kterém se hledají diskuzní skupiny. Tento tág se může vyskytovat opakovaně.
`timeout`	Čas v milisekundách, do kterého musí server při připojení odpovědět, než je označen za nedostupný.
`repeat`	Pokud se nepodaří připojit, pak hodnota zde uvedená definuje počet pokusu připojení k jednomu serveru.
`reconnect`	Prodleva v sekundách mezi opětovnými pokusy připojení k témuž serveru.
`delay`	Prodleva mezi aktualizacemi zpráv na NNTP serveru. Uvádí se v minutách.
`filter`	Tento tág může obsahovat dva synovské prvky: `permit` a `deny`. Do těchto se uvádí oslabené regulární výrazy, pro povolení diskuzních skupin, které se budou kontrolovat.

Oslabení regulárních výrazu v tomto případě znamená, že symboly tečka (`.`) a hvězdička (`*`) nemají původní významy. Tečka představuje jen tečku a hvězdička znamená libovolný název diskuzní skupiny.

Základní politika je zakázat vše, co není explicitně povoleno. Není-li tedy uveden žádný element `permit`, nestáhne se ani jedna zpráva. Dalo by se říci, že přítomnost elementu `deny` je zde zbytečná. Avšak je zde použita pro větší pohodlí konfigurace.

Je nutno podotknout, že pravidla `deny` se uplatňují vždy až naposled. Tento způsob aplikace pravidel připouští povolit několik diskuzních skupin z nadřazené kategorie, s explicitní výjimkou několika konkrétních, aniž by bylo nutné vypisovat všechny. Viz následující příklad, který slouží k povolení všech diskuzních skupin `vutbr.fit.*` ale nepovoluje podkategorii `courses`.

```
<filter>
<permit>vutbr.fit.*</permit>
<deny>vutbr.fit.courses.*</permit>
</filter>
```

Lokální nastavení

`authent` Pokud je tato položka obsažena, musí obsahovat alespoň jeden element `server`. Seznam těchto serveru je uživateli nabídnut při registraci. O proti vybranému serveru je autentizován. Pokud není tento prvek přítomen, pak se mohou registrovat libovolní uživatelé.

`debug` Přítomnost tohoto elementu způsobí, že se na standardní výstup vypisuje probíhající komunikace s XMPP i NNTP serverem. Pokud se v tomto elementu uvede klíčové slovo `XMPP` nebo `NNTP`, je vypisována jen jedna z těchto komunikací.

Jazyková konfigurace

Jazyková konfigurace je v současné době jen v základní podobě, ale její podoba je již teď nastavena tak, aby ji v budoucnu bylo možné rozšířit pro podporu více jazyků, kdy se jazyk odpovědí bude volit podle nativního jazyka uživatele.

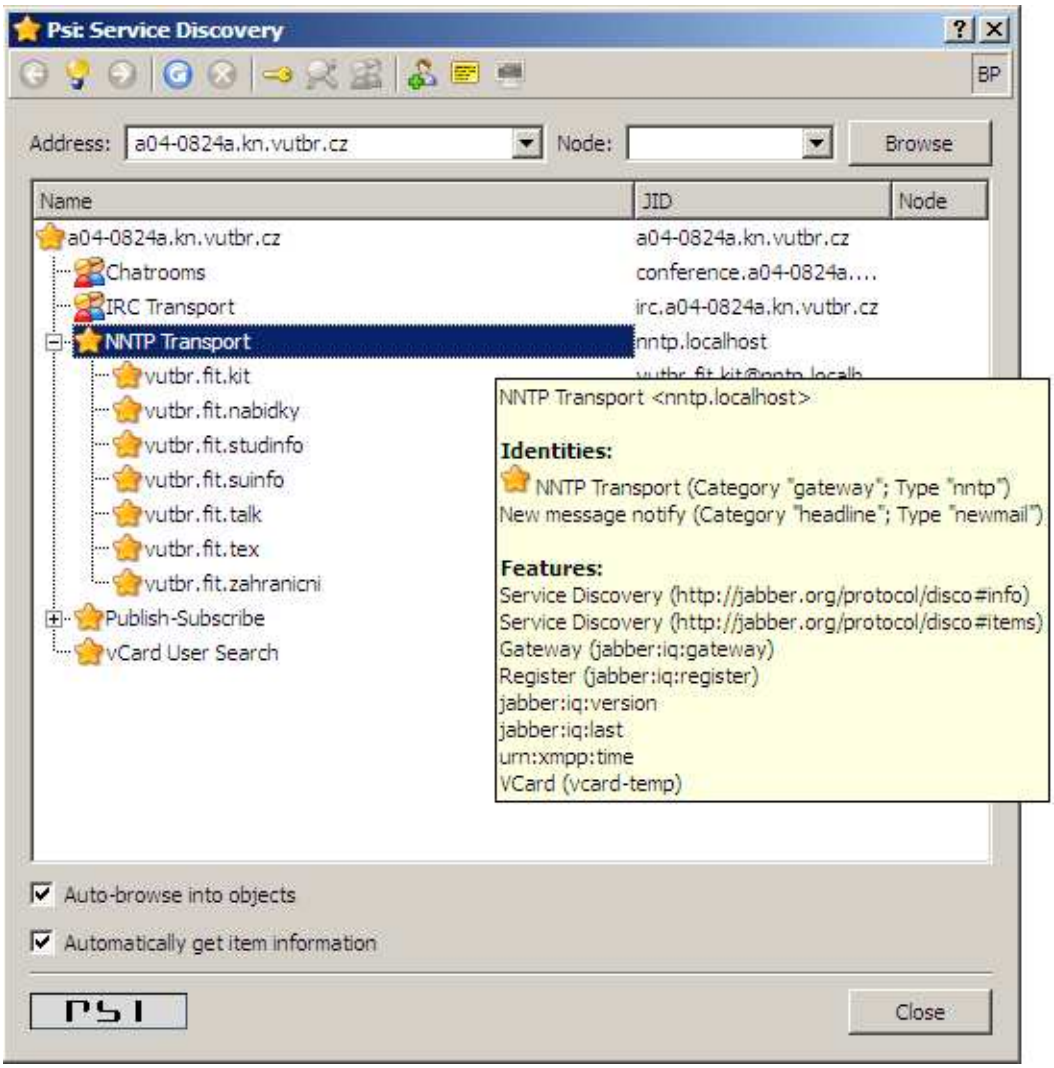
Obsah jednotlivých elementů tvoří zprávy, jenž se zasílají uživateli. Co který element znamená není nutné představovat a lze to odvodit překladem z angličtiny.

Kořenový element této části má parametr `lang`, který určuje pro který jazyk je konfigurován.

V současné době se tato část může vyskytovat jen jedenkrát, a nezáleží na parametru lang.

3.3 Ovládání

Jakmile je nastaven konfigurační soubor a služba spuštěna, již není třeba dalších zásahů. Pokud uživatel chce této služby využívat, postupuje podle kroků popsáných v následujících kapitolách.



Obrázek 3.2: Service Discovery v klientovi Psi v0.11

3.4 Správa odebíraných skupin

1. Zobrazit seznam nabízených služeb (*Service Discovery*)
2. Zaregistrovat se (je-li to vyžadováno)
3. Přidat si požadované skupiny do resteru

Přidáním skupiny, se uživatel automaticky přihlašuje k odběru nových zpráv (událost „trap“). Pokud již uživatel nechce tuto skupinu dále odebírat, tak tento kontakt jednoduše odstraní ze svého *rosteru*.

3.4.1 Čtení příspěvku

Zprávy je možné dostávat dvěma způsoby: automaticky, když se objeví nová zpráva na serveru, a na vyžádání. Aby bylo možné si zprávu vyžádat, je nejprve nutné vědět, jaké zprávy na serveru jsou. K tomuto účelu slouží příkaz `list`. Tento, i ostatní příkazy se zasílají konkrétní skupině, reprezentované kontaktem, jako prostá textová zpráva. Reakci na uvedený příkaz je zaslání tabulky všech dostupných zpráv, resp. jejích hlaviček (viz Obrázek 3.3).

```
ID     Předmět
       Datum
       Od
-----------------------------------------------
1      curiculum‘s v 4.0
       Tue, 19 Feb 2008 23:28:55
       Jan Novák <xnovak99@stud.fit.vutbr.cz>

2      Re: curiculum‘s v 4.0
       Wed, 20 Feb 2008 08:03:20
       Josef Novotný <novotny@fit.vutbr.cz>
```

Obrázek 3.3: Transport IM služeb

Pro další manipulaci se zprávami je nutné číslo uvedeno jako *ID*. Toto číslo tvoří argument příkazu `GET`, který se zasílá vždy s parametrem. Ten se odděluje mezerou. Na příkaz se správným parametrem, odpoví server zasláním celé zprávy.

3.4.2 Přidávání příspěvku

Zasílaní příspěvků na server je poněkud složitější operací. Cílem je vytvořit uživatelský přístupnou aplikaci (nezávislou na klientech), a tak není možné spoléhat na všechny dostupné vlastnosti protokolu XMPP. Například u klienta připojujícího se přes *Mirandu* není vždy zaručeno, že podporuje zprávy typu `normal`. (Podlé mých zkušeností je umí přijímat, ale nemá formulář, ve kterém by se dalo využit pole *subject* při posílání zpráv). Z tohoto důvodu bylo vybráno řešení, které se může zdát pro uživatele poněkud komplikovanější, ale na druhou stranu je vždy použitelné.

Zprávy se tedy na server zasílají v následující podobě `POST název zprávy EOL text zprávy`.

Příklad: Zaslání zprávy na server, jejíž předmětem je „Testovací zpráva“ a zdělení zprávy zní: „Příliš žluťoučký kůň úpěl ďábelské ódy.“.

```
POST Testovací zpráva
Příliš žluťoučký kůň úpěl ďábelské ódy.
```

Zpráva je tedy sestavena tak, že za klíčovým slovem `POST` následuje předmět zprávy, který se zakončí koncem řádku (obvykle se vkládá pomocí kombinace kláves

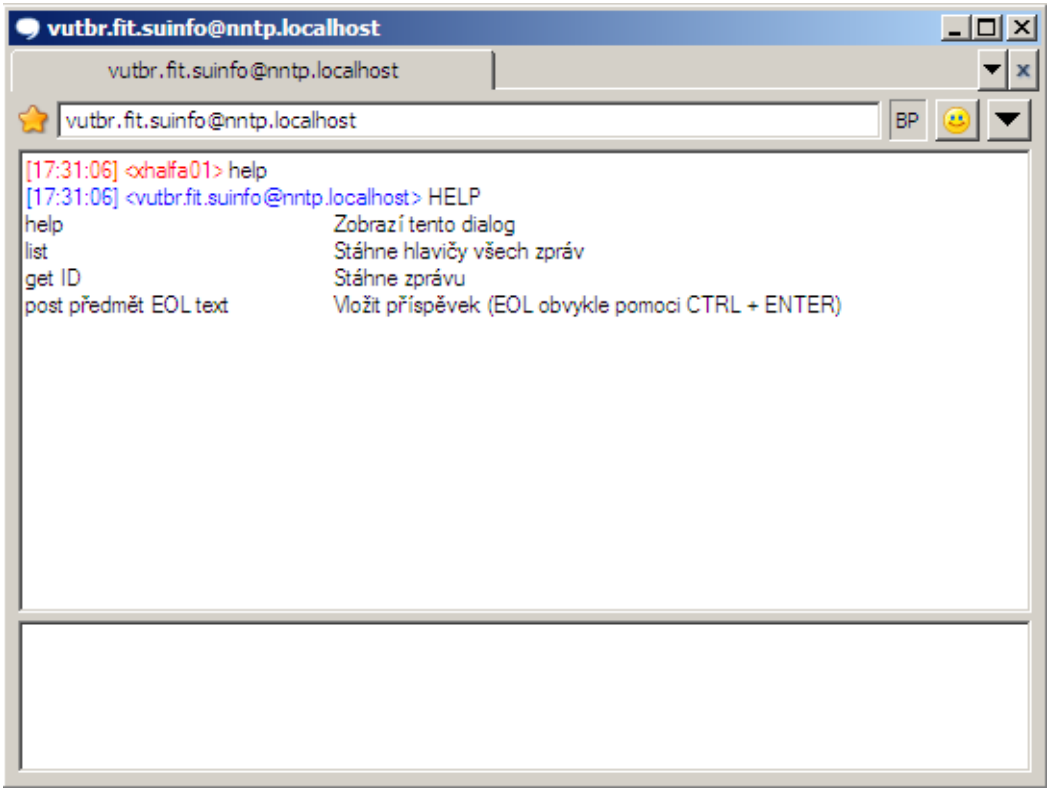
CTRL + ENTER). Na dalším řádku začíná obsah těla zprávy. Další zalomení řádku nejsou brána v zřetel.

Zaslané příspěvky se nenahrávají na *NetNews* server okamžitě, nýbrž se ukládají do XML souboru. K nahrání příspěvků na server dochází v rámci aktualizace zpráv.

Ještě před tím, než se zpráva uloží do XML souboru, prochází předzpracováním. Jeho cílem je upravit řádky, které přesahují délku 80ti znaků. Dojde tedy k zalomení řádku v místě poslední mezery před 78 znakem.

3.4.3 Nápověda

Pokud uživatel neví, jaké příkazy je možné použít, nebo jaké parametry jsou u kterého příkazu použity, stačí serveru zaslat libovolný text. Na neznámou zprávu se odpovídá informací o možnosti využití příkazu `HELP`. V odpovědi na toto klíčové slovo server zašle uživateli seznam a tvar všech možných příkazu, včetně jejích popisu. Příklad je na obrázku 3.4.



Obrázek 3.4: Výpis nápovědy

3.5 Rozšíření

Rozšíření, které lze k této službě doplnit je bezpočet, a omezení je dáno především ze strany uživatelských klientů (záleží na podpoře rozšiřujících protokolů *XEP*). Některé

ze zde uvedených návrhu jsou již implementovány, některé budou během tisku této zprávy, další se objeví před konečným vydáním této aplikace k volnému použití a poslední budou přidávány do dalších verzí.

3.5.1 Kódování zpráv

Již v úvodu bylo uvedeno, že pro ukládání *NetNews* zpráv se používá kódování *US-ASCII*. Protože většina klientů se snaží uživateli nabídnout možnost používat i národní znaky, používají se standardy, které tyto možnosti umožňují, jedná se zejména o *MIME – Multipurpose Internet Mail Extensions*, které dovoluje zakódovat nejen obsah těla zprávy, ale i jednořádkové položky, jaké jsou například v hlavičce zpráv (Subject, From, aj.).

Způsob kódování polí hlavičky je popsán v *RFC 2047 – MIME Part Three: Message Header Extensions for Non-ASCII Text*[16].

Tělo zprávy již je částečně podle rozšíření *MIME* zpracováváno, neboť se vyhodnocuje, zda zpráva neobsahuje přílohu. V této situaci je další zpracování závislé jen na zpracování parametrů a následném překódování.

3.5.2 Formuláře

Formuláře jsou již v aplikaci částečně nasazeny, a to při registraci služby, kdy se po uživateli vyžaduje zadání některých parametrů.

V tomto případě se však jedná o rozšíření zpráv, kdy se místo běžných textových zpráv dá využít právě formulářů. Toto rozšíření však naráží na nižší podporu u klientů.

Zajímavé by mohlo býti i rozšíření samotného protokolu *XEP-0004*, například o objekt podporující stromové zobrazení.

3.5.3 Formát zobrazení

Další možnosti, jak zobrazovat seznam příspěvků, je jejích stromová struktura.Položka *References* je uchovávána, takže závislosti jsou dostupné. Toto rozšíření by mohlo být dostupné v některé z dalších verzí.

Otázkou je, zda implementovat jako samostatný příkaz, nebo jako parametr příkazu `list`. V případě druhé varianty, by bylo možné aby si uživatel nastavil implicitní chování samotného příkazu list podle svých představ.

3.5.4 Přílohy

V současné době je stav takový, že program dokáže rozpoznat, přílohy ve zprávě a a záleží na nastavení, zda si je uloží, či ne. Je doporučeno, v současné době zakázat ukládání příloh, neboť je nelze dále zpracovávat. I v tomto režimu je uživatel informován o názvu a datovém typu přílohy, avšak v případě zájmu o tuto přílohu se musí připojit přímo k *NetNews* serveru, aby si ji mohl stáhnout.

V tomto ohledu se nabízí vytvořit rozšíření pro podporu protokolu *XEP-0096* a zavedení nového klíčového slova (např. `download <msg_id> <attach_id>`. Na tento příkaz by server reagoval pokusem o zaslání přílohy číslo `<attach_id>` ze zprávy identifikované pomocí `<msg_id>`.

3.5.5 Vlákna

V současné době není vyřešen způsob odpovědi na zprávu tak, aby vy vytvořila závislost. Jako ideální možnost se jevilo využití vlastnosti XMPP protokolu, který dovoluje definovat konverzační vlákna, čímž by odpadla veškerá režie pro uživatele. Nicméně tato vlastnost je definována jako volitelná, a tak není podporována ze strany klientů.

Aby bylo možno odpovídat na příspěvky, bude tato vlastnost doplněná buďto ještě před vydáním první verze, nebo do některé bezprostředně následující. A to zavedením nového příkazu, jehož parametrem bude ID zprávy, na kterou se odpovídá.

Závěr

Doba si vyžaduje kontrolu nad informacemi. Tato bakalářská práce ji nemůže poskytnout v plném rozsahu, ale snaží se nabídnout možnosti, které by mohly z dosavad méně využívaných služeb vytvořit konkurenci současně se rozmáhajícím webovým službám.

V první části této práce byly popsány dva protokoly. Jeden docela starší, a dnešní mladé počítačové komunitě možná i neznámy, a druhý nový a slibně se rozvíjející projekt. Ve zbývající části je nastíněna možnost propojení těchto dvou protokolů, která je detailněji komentována na vytvořené aplikaci.

Problematika obou protokolů i navrženého řešení je mnohém sofistikovanější, a není v možnostech této bakalářské práce ji zahrnout kompletně. Všechny informace potřebné k nastudováni teto problematiky, lze nalézt v materiálech, které jsou uvedeny v seznamu literatury.

Důraz je v této práci kladen na popis vytvořené aplikace a její ovládání. Další vývoj aplikace je otevřen, a měla by mu pomoci dostupná dokumentace kódu, která je součásti přiloženého CD.

Literatura

[1] David H. Crocker. Standard for the format of arpa internet text messages. Technical Report RFC 822, University of Delaware, 1982.

[2] Brian Kantor (U.C. San Diego) and Phil Lapsley (U.C. Berkeley). Network news transfer protocol. Technical Report RFC 977, 1986.

[3] C. Feather. Network news transfer protocol (nntp). Technical Report RFC 3977, THUS plc, 2006.

[4] Elliotte Rusty Harrold and W. Scott Means. *XML v kostce: pohotová referenční příručka*. Computer Press, 2002. ISBN 8072267124.

[5] Mark R. Horton. Standard for interchange of usenet messages. Technical Report RFC 850, 1983.

[6] T. Dierks (Independent) and E. Rescorla (RTFM, Inc.). The transport layer security (tls) protocol version 1.1. Technical Report RFC 4346, 2006.

[7] N. Freed (Innosoft), J. Klenstin (MCI), and J. Postel (ISI). Multipurpose internet mail extensions (mime) part four: Registration procedures. Technical Report RFC 2048, 1996.

[8] N. Freed (Innosoft) and N. Borenstein (First Virtual). Multipurpose internet mail extensions (mime) part five: Conformance criteria and examples. Technical Report RFC 2049, 1996.

[9] N. Freed (Innosoft) and N. Borenstein (First Virtual). Multipurpose internet mail extensions (mime) part one: Format of internet message bodies. Technical Report RFC 2045, 1996.

[10] N. Freed (Innosoft) and N. Borenstein (First Virtual). Multipurpose internet mail extensions (mime) part two: Media types. Technical Report RFC 2046, 1996.

[11] M. Horton (AT&T Bell Laboratories) and R. Adams (Center for Seismic Studies). Standard for interchange of usenet messages. Technical Report RFC 1036, 1987.

[12] A. Melnikov, Ed. (Isode Limited) and K. Zeilenga, Ed. (OpenLDAP Foundation). Simple authentication and security layer (sasl). Technical Report RFC 4422, 2006.

[13] M. Day (Lotus), J. Rosenberg (dynamicsoft), and H. Sugano (Fujitsu). A model for presence and instant messaging. Technical Report RFC 2778, 2000.

[14] M. Day (Lotus), S. Aggarwal (Microsoft), G. Mohr(Activerse), and J. Vincent (Into Networks). Instant messaging / presence protocol requirements. Technical Report RFC 2779, 2000.

[15] A. Mayrhofer. Iana registration for enumservice 'xmpp'. Technical Report RFC 4979, enum.at, 2007.

[16] K. Moore. Mime (multipurpose internet mail extensions) part three: Message header extensions for non-ascii text. Technical Report RFC 2047, University of Tennessee, 1996.

[17] P. Resnick, Editor. Internet message format. Technical Report RFC 2822, QUALCOMM Incorporated, 2001.

[18] P. Saint-Andre, Ed. Extensible messaging and presence protocol (xmpp): Core. Technical Report RFC 3920, Jabber Software Foundation, 2004.

[19] P. Saint-Andre, Ed. Extensible messaging and presence protocol (xmpp): Instant messaging and presence. Technical Report RFC 3921, Jabber Software Foundation, 2004.

[20] P. Saint-Andre. End-to-end signing and object encryption for the extensible messaging and presence protocol (xmpp). Technical Report RFC 3923, Jabber Software Foundation, 2004.

[21] P. Saint-Andre. Mapping the extensible messaging and presence protocol (xmpp) to common presence and instant messaging (cpim). Technical Report RFC 3922, Jabber Software Foundation, 2004.

[22] P. Saint-Andre. Internationalized resource identifiers (iris) and uniform resource identifiers (uris) for the extensible messaging and presence protocol (xmpp). Technical Report RFC 4622, JSF, 2006.

[23] P. Saint-Andre. A uniform resource name (urn) namespace for extensions to the extensible messaging and presence protocol (xmpp). Technical Report RFC 4854, XSF, 2007.

[24] P. Saint-Andre. Internationalized resource identifiers (iris) and uniform resource identifiers (uris) for the extensible messaging and presence protocol (xmpp). Technical Report RFC 5122, XSF, 2008.

[25] Michael Tobler. *Inside Linux*, chapter 17 Internet News Service. Sams Publishing, 2000. ISBN 0735709408.

Seznam použitých zkratek

ARPA	Advanced Research Projects Agency
ASCII	American Standard Code for Information Interchange
CD	Compact Disc
CPIM	Common Presence and Instant Messaging
CR	Carriage Return
DNS	Domain Name Service
EOL	End of Line
IANA	Internet Assigned Numbers Authority
ID	Identification
IETF	Internet Engineering Task Force
IM	Instant Messaging
IP	Internet Protocol
IRC	Internet Relay Chat
IRI	International Resource Identifiers
JID	Jabber ID
JSF	Jabber Software Foundation
LF	Line Feed
MIME	Multipurpose Internet Mail Extensions
NNTP	Network News Transfer Protocol
RSS	Really Simple Syndication
SASL	Simple Authentication and Security Layer
SMTP	Simple Mail Transfer Protocol
TLS	Transport Layer Security
URI	Uniform Resource Identifier
URN	Uniform Resource Name
USENET	User Network
VIP	Very Important Person
XEP	XMPP Extension Protocol
XML	Extensible Markup Language
XMPP	Extensible Messaging and Presence Protocol

Příloha A

Konfigurace serveru

Tato příloha obsahuje popis konfigurace pro nejpoužívanější jabber servery.

Použité identifikátory:

__JID__ představuje Jabber ID, pod kterým je služba dostupna. (např. nntp.localhost)

_shared_key_ je sdílené heslo. Stejná fáze musí být uvedena v konfiguračním souboru NNTP Transportu (*config.xml*).

A.1 Jabberd14

Níže uvedené části přijdou vložit do souboru `jabber.xml`, který se nachází ve složce, do níž byl jabber server nainstalován.

První část přijde připojit do uzlu */jabber/service[id='session']/jsm/browse*.

```
<service type="nntp" jid="__JID__" name="NNTP Transport">
  <ns>jabber:iq:version</ns>
  <ns>jabber:iq:register</ns>
</service>
```

Druhá část se umístí do stejného souboru, jako potomek */jabber/*.

```
<service id="__JID__">
  <accept>
    <ip/>
    <port>5232</port> <!-- 5232 is default -->
    <secret>_shared_key_</secret>
  </accept>
</service>
```

A.2 Ejabberd

Konfigurace tohoto serveru se provádí v souboru `ejabberd.cfg`, který je umístěn v podadresáři `conf` instalační složky. Níže uvedené řádky přijdou připsat do části *LISTENING PORTS*.

```
{5232, ejabberd_service, [  %%(5232 is default)
    {access, all},
    {shaper_rule, fast},
    {hosts, ["__JID__"],
      [{password, "_shared_key_"}]}
    ]},
```

Takto se upraví položka *mod_disco* v tomtéž dokumentu v části *MODULES*.

```
{mod_disco, [{extra_domains, ["__JID__"]}]},
```

Příloha B

Schémata

Tato příloha obsahuje XML schémata pro XML soubory, využívané implementovaným softwarem.

B.1 config.xsd

Příloha obsahuje schéma konfiguračního souboru `config.xml`. Tento soubor se nalézá v instalačním adresáři. Popis sémantického významu uzlů je uveden v kapitole 3.2.4 na stránce 20.

```
<?xml version="1.0" ?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<!-- named restrictions -->
<xs:complexType name="empty"/>
<xs:simpleType name="mail">
  <xs:restriction base="xs:string">
    <xs:pattern value="[a-zA-Z0-9]+@[a-zA-Z0-9]+"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="debug_mode">
  <xs:restriction base="xs:string">
    <xs:enumeration value=""/>
    <xs:enumeration value="XMPP"/>
    <xs:enumeration value="NNTP"/>
  </xs:restriction>
</xs:simpleType>

<!-- attribute -->
<xs:attribute name="lang" type="xs:string"/>
```

```
<!-- simple elements -->
<xs:element name="jid" type="mail"/>
<xs:element name="server" type="xs:string"/>
<xs:element name="port" type="xs:positiveString"/>
<xs:element name="shared_key" type='xs:string'/>
<xs:element name="timeout" type="xs:positiveInteger"/>
<xs:element name="reconnect" type="xs:positiveInteger"/>
<xs:element name="show_item" type="empty"/>

<xs:element name="FN" type="xs:string"/>
<xs:element name="GIVEN" type="xs:string"/>
<xs:element name="FAMILY" type="xs:string"/>
<xs:element name="URL" type="xs:string"/>
<xs:element name="EMAIL" type="mail"/>
<xs:element name="JABBERID" type="mail"/>
<xs:element name="DESC" type="xs:string"/>

<xs:element name="delay" type="xs:positiveInteger"/>
<xs:element name="repeat" type="xs:positiveInteger"/>
<xs:element name="no_attachment" type="empty"/>

<xs:element name="permit" type="xs:string"/>
<xs:element name="deny" type="xs:string"/>

<xs:element name="title" type="xs:string"/>
<xs:element name="instruction" type="xs:string"/>
<xs:element name="name" type="xs:string"/>
<xs:element name="password" type="xs:string"/>
<xs:element name="from" type="xs:string"/>
<xs:element name="date" type="xs:string"/>
<xs:element name="subject" type="xs:string"/>
<xs:element name="attach" type="xs:string"/>
<xs:element name="unknown" type="xs:string"/>
<xs:element name="help" type="xs:string"/>
<xs:element name="list" type="xs:string"/>
<xs:element name="get" type="xs:string"/>
<xs:element name="post" type="xs:string"/>
<xs:element name="badParam" type="xs:string"/>

<xs:element name="debug" type="debug_mode"/>

<!-- complex type -->
<xs:element name="register">
<!-- Labels for registration form -->
  <xs:complexType>
```

```
    <xs:all>
      <xs:element ref="title"/>
      <xs:element ref="instruction"/>
      <xs:element ref="name"/>
      <xs:element ref="password"/>
      <xs:element ref="server"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="message">
<!-- Captions of message header's elements -->
  <xs:complexType>
    <xs:all>
      <xs:element ref="from"/>
      <xs:element ref="date"/>
      <xs:element ref="subject"/>
      <xs:element ref="attach"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="commands">
<!-- Description of commans for help -->
  <xs:complexType>
    <xs:all>
      <xs:element ref="unknown"/>
      <xs:element ref="help"/>
      <xs:element ref="list"/>
      <xs:element ref="get"/>
      <xs:element ref="post"/>
      <xs:element ref="badParam"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="authent">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="server" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="filter">
  <xs:complexType>
    <xs:choice>
      <xs:element ref="permit" minOccurs="1" maxOccurs="unbounded"/>
      <xs:element ref="deny" minOccurs="0" maxOccurs="unbounded"/>
```

```
    </xs:choice>
  </xs:complexType>
</xs:element>
<xs:element name="N">
  <xs:complexType>
    <xs:all>
      <xs:element ref="GIVEN"/>
      <xs:element ref="FAMILY"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="vCard">
  <xs:complexType>
    <xs:all>
      <xs:element ref="FN"/>
      <xs:element ref="N"/>
      <xs:element ref="URL"/>
      <xs:element ref="EMAIL"/>
      <xs:element ref="JABBERID"/>
      <xs:element ref="DESC"/>
    </xs:all>
  </xs:complexType>
</xs:element>

<xs:element name="xmpp">
  <xs:complexType>
    <xs:all>
      <xs:element ref="jid"/>
      <xs:element ref="server"/>
      <xs:element ref="port"/>
      <xs:element ref="shared_key"/>
      <xs:element ref="timeout"/>
      <xs:element ref="reconnect"/>
      <xs:element ref="show_item" minOccurs="0"/>
      <xs:element ref="vCard"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="nntp">
  <xs:complexType>
    <xs:all>
      <xs:element ref="server"/>
      <xs:element ref="timeout"/>
      <xs:element ref="reconnect"/>
      <xs:element ref="delay"/>
```

```
      <xs:element ref="repeat"/>
      <xs:element ref="filter"/>
      <xs:element ref="no_attachment"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="locale">
  <xs:complexType>
    <xs:all>
      <xs:element ref="debug" minOccurs="0"/>
      <xs:element ref="authent" minOccurs="0"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="language">
  <xs:complexType>
    <xs:all>
      <xs:element ref="register"/>
      <xs:element ref="message"/>
      <xs:element ref="commands"/>
    </xs:all>
    <xs:attribute ref="lang" use="required"/>
  </xs:complexType>
</xs:element>

<xs:element name="configure">
  <xs:complexType>
    <xs:all>
      <xs:element ref="xmpp"/>
      <xs:element ref="nntp"/>
      <xs:element ref="locale"/>
      <xs:element ref="language"/>
    </xs:all>
  </xs:complexType>
</xs:element>

</xs:schema>
```

B.2 news.xsd

Schéma popisuje soubor `news.xml`. Neexistuje-li, pak se vytvoří automaticky. Nevyžaduje žádné zásahy.

```
<?xml version="1.0" ?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<!-- named restrictions -->
<xs:complexType name="empty"/>
<xs:simpleType name="code_type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="7bit"/>
    <xs:enumeration value="quoted-printable"/>
    <xs:enumeration value="base64"/>
    <xs:enumeration value="8bit"/>
    <xs:enumeration value="binary"/>
  </xs:restriction>
</xs:simpleType>

<!-- attribute -->
<xs:attribute name="name" type="xs:string"/>
<xs:attribute name="type" type="xs:string"/>
<xs:attribute name="id" type="xs:positiveInteger"/>
<xs:attribute name="encoding" type="code_type"/>

<!-- simple elements -->
<xs:element name="reference" type="xs:string"/>
<xs:element name="msg-id" type="xs:string"/>
<xs:element name="to" type="xs:string"/>
<xs:element name="from" type="xs:string"/>
<xs:element name="date" type="xs:string"/>
<xs:element name="subject" type="xs:string"/>
<xs:element name="body" type="xs:string"/>
<xs:element name="new" type="empty"/>
<xs:element name="delete" type="empty"/>

<xs:element name="attach">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute ref="name" use="required"/>
        <xs:attribute ref="type" use="required"/>
        <xs:attribute ref="encoding"/>
```

```
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<!-- complex type -->
<xs:element name="attachment">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="attach" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="head">
  <xs:complexType>
    <xs:all>
      <xs:element ref="from"/>
      <xs:element ref="subject"/>
      <xs:element ref="date"/>
      <xs:element ref="msg-id"/>
      <xs:element ref="reference" minOccurs="0"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="message">
  <xs:complexType>
    <xs:all>
      <xs:element ref="head"/>
      <xs:element ref="attachment" minOccurs="0"/>
      <xs:element ref="body"/>
      <xs:element ref="new" minOccurs="0"/>
      <xs:element ref="delete" minOccurs="0"/>
    </xs:all>
    <xs:attribute ref="id" use="required"/>
  </xs:complexType>
</xs:element>
<xs:element name="group">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="message" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="name" use="required"/>
  </xs:complexType>
  <xs:unique name="id_unique">
    <xs:selector xpath="message"/>
```

```
    <xs:field xpath="@id"/>
  </xs:unique>
</xs:element>
<xs:element name="msg">
  <xs:complexType>
    <xs:all>
      <xs:element ref="to"/>
      <xs:element ref="from"/>
      <xs:element ref="subject"/>
      <xs:element ref="body"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="post">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="msg" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="newsgroups">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="group" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="post" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

B.3 users.xsd

Schéma pro soubor `users.xml`, v němž se uchovávají informace o všech registrovaných uživatelích. Uzel *subscribe* obsahuje seznam odebíraných diskuzních skupin. Soubor je modifikován softwarem a nevyžaduje zásahy od správce.

```
<?xml version="1.0" ?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<!-- named restrictions -->
<xs:simpleType name="mail">
  <xs:restriction base="xs:string">
    <xs:pattern value="[a-zA-Z0-9]+@[a-zA-Z0-9]+"/>
  </xs:restriction>
</xs:simpleType>

<!-- attribute -->
<xs:attribute name="jid" type="mail"/>

<!-- simple elements -->
<xs:element name="name" type="xs:string"/>
<xs:element name="server" type="xs:string"/>
<xs:element name="login" type="xs:string"/>
<xs:element name="group" type="xs:string"/>


<!-- complex type -->
<xs:element name="detail">
  <xs:complexType>
    <xs:all>
      <xs:element ref="login"/>
      <xs:element ref="server"/>
      <xs:element ref="name"/>
    </xs:all>
  </xs:complexType>
</xs:element>
<xs:element name="subscribe">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="group" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="user">
```

```
  <xs:complexType>
    <xs:all>
      <xs:element ref="detail"/>
      <xs:element ref="subscribe"/>
    </xs:all>
    <xs:attribute ref="jid" use="required"/>
  </xs:complexType>
</xs:element>

<xs:element name="configure">
  <xs:complexType>
    <xs:all>
      <xs:element ref="user"/>
    </xs:all>
  </xs:complexType>
</xs:element>

</xs:schema>
```

Příloha C

Obsah CD

- Bakalářská práce ve formátu PDF
- Zdrojové kódy aplikace
- Dokumentace zdrojových kódu