

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

DATAMINING Z JABBERU

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JAROSLAV SENDLER

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

DATAMINING Z JABBERU

DATAMINING FROM JABBER

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

JAROSLAV SENDLER

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. JOZEF MLÍCH

BRNO 2011

Abstrakt

Předmětem této bakalářské práce bylo seznámení se s problematikou komunikace přes Jabber síť, která zde byla rozebrána. Konkrétním cílem bylo vytvoření jednoduchého Jabber klienta, který by byl schopen získávat statistická data. Nashromážděná data sloužila pro pozdější analýzu a grafickou reprezentaci informací z nich získaných.

Abstract

The objective of this thesis was acquaint oneself with problems of communication via Jabber network, which was also analyzed. The specific objective was to create a simple Jabber's client which would be able to obtain statistical data. The collected data was used for analysis and graphic representation of information.

Klíčová slova

Jabber, XMPP, robot, datamining, dolování dat, RapidMiner.

Keywords

Jabber, XMPP, robot, datamining, RapidMiner.

Citace

Jaroslav Sendler: Datamining z jabberu, bakalářská práce, Brno, FIT VUT v Brně, 2011

Datamining z jabberu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jozefa Mlícha. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....

Jaroslav Sendler

9. května 2011

Poděkování

Tímto bych chtěl poděkovat mému vedoucímu bakalářské práce Ing. Jozefovi Mlíchovi za ochotu a kladný přístup při konzultacích. Dále za poskytnutí hardware na němž běžel program a sbíral data.

© Jaroslav Sendler, 2011.

Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

1 Úvod	2
2 XMPP	4
2.1 Architektura	4
2.2 XML	7
2.3 Stanza	8
2.4 Rozšíření	10
3 Data mining	13
3.1 Transformace dat	16
3.2 Metody dolování dat	19
3.3 Shlukování	20
3.4 Programy	23
4 Implementace	25
4.1 Architektura	25
4.2 Databáze	26
4.3 Robot	28
5 Pokračování práce	31
6 Vyhodnocení výsledků	33
6.1 Manuální rozbor dat	33
6.2	33
7 Závěr	34
A Obsah CD	38
B Manual	39
C Konfigurační soubor	40
D Slovník zkratk	41
E Stanza - základní schéma	42
F Návrh databáze	47

1 Kapitola 1

2 Úvod

3 V současné době jsou řazeny k nejrozšířenějším psaným dorozumívacím prostředkům real-
4 time komunikační sítě. Samozřejmě je tento jev k vidění až koncem minulého desetiletí.
5 Psaná komunikace je již několik století využívána jako prostředek k dorozumívání lidí mezi
6 sebou. Za toto období prošla výrazným pokrokovým vývojem, proto u ní lze nalézt mnoho
7 časových mezníků, které ji výrazně ovlivnily. Jako příklad jednoho z prvních komunikačních
8 kanálů lze uvést posly, kteří často nesly zprávy na vzdálenosti několika kilometrů. Dalším
9 výrazným prvkem ve vývoji dorozumívacích prostředků bylo zavedení pošty a objevení
10 telegrafu.

11 Příchodem internetu nastal v komunikaci zásadní zlom. Postupným rozšířením pokrytí
12 a dostupnosti této technologie začalo vznikat mnoho nových komunikačních prostředků.
13 Příkladem mohou být elektronické zprávy, RSS zprávy nebo real-time komunikační sítě.
14 Právě poslední zmíněná metoda zažívala v moderní době velký růst v oblasti popularity,
15 ať už v podobě ICQ protokolu nebo dnes velmi rozšířené sociální sítě Facebook. Jedna z
16 hlavních výhod těchto služeb, například v porovnání s klasickou poštou, je jejich rychlost
17 doručení. Naproti běžné elektronické poště jsou uživatelé informováni o stavu příjemce
18 zprávy. Real-time komunikační prostředky nabízejí služby, kterými je možné zjistit zda
19 se příjemce zprávy nachází u dorozumívacího zařízení. Díky této schopnosti je uživateli
20 umožněno zasílat zprávy a obratem na ně očekávat odpovědi.

21 S přibývajícími elektronickými daty, na poli ať už vědních nebo praktických oborů,
22 přichází potřeba tyto informace uchovávat. K těmto účelům slouží databáze, které se svou
23 strukturou zaměřují především na snadnou kontrolu dat, vyhledávání a jejich analyzování.
24 S rostoucím obsahem se ale uložené informace stávají pouze daty bez významu. Proto
25 vnikla nová disciplína nazvaná *data mining*, která si klade za cíl znovunalezení „ztracených“
26 informací (na první pohled neviditelných) nebo nalezení informací úplně nových.

27 Předmětem této bakalářské práce je seznámení se s problematikou komunikace probíha-
28 jící přes Jabber síť. Konkrétním cílem je vytvoření jednoduchého Jabber klienta, který by
29 byl schopen získávat statistická data. Nashromážděná data jsou dále využita pro pozdější
30 analýzu a grafickou reprezentaci informací z nich získaných. Hlavním cílem této práce je
31 získat neznámé informace z real-time komunikační sítě Jabber.

32 Bakalářská práce je tvořena z šesti kapitol. Kapitola **druhá** je tvořena popisem proto-
33 kolu XMPP, na kterém je postavena real-time komunikační služba Jabber. Je zde popsána
34 architektura sítě a základní stavební kameny XMPP protokolu, které jsou využívány sítí
35 Jabber jako nástroj k zprostředkování jednotlivých služeb. V závěru tohoto oddílu je vě-
36 nována část vybraným standardům, které rozšiřují základní XMPP protokoly. Jsou zde
37 uvedena pouze ta rozšíření, která byla po konzultaci s vedoucím práce, označena za rele-

38 vantní k této práci.

39 Obsah **třetí** kapitoly je zaměřen na popis procesu data mining. Zabývá se začleněním
40 této metody do komplexnějšího procesu, který je nazýván získávání znalostí z databází. Da-
41 lší součást této kapitoly se zabývá nezbytnými kroky transformace dat, která jsou získána
42 z Jabber komunikace. Při tomto procesu jsou data převedena do vhodné podoby pro data
43 minig. Následuje část, kterou jsou popsány vybrané metody dolování dat a také je zde po-
44 drobně rozebrán algoritmus k -means. Tento algoritmus je využit pro samotné dolování dat,
45 které je zprostředkováno aplikací RapidMiner. Popis tohoto programu a dalších vybraných
46 nástrojů uzavírá třetí kapitolu.

47 Implementační část této práce je popsána kapitolou **čtvrtou**. Je členěna na oddíly,
48 které odpovídají vybraným částem architektury této práce. Elementy architektury, které
49 jsou popsány vlastní podkapitolou, jsou rozšířeny o zjednodušený návrh struktury v podobě
50 grafických schémat.

51 Kapitola **pátá** se zamýšlí nad rozšířeními robota a nad dalším pokračováním této práce.
52 Výsledky získané touto prací jsou prezentovány v kapitole **šesté**. Je zde ukázán model pou-
53 žívaný programem RapidMiner pro dobývání znalostí. Konkrétně jde o shlukování uživatelů
54 sítě Jabber do skupin podle času stráveném v samotné síti a jejich stavu. Výsledky jsou
55 představeny pomocí různých grafických entit, ať už v podobě diagramu nebo grafu, který
56 je transformovaný do dvourozměrného prostoru.

57 Nemalý informativní obsah tvoří i **přílohy**. V první řadě to je slovník zkratk, shrnující
58 slovní spojení z celého tohoto dokumentu. Následuje podrobnější popis základních entit
59 stanzy a přehled průběhu rozšíření. V další části je ukázán kompletní návrh databáze,
60 který doplňuje základní popis ze čtvrté kapitole.

61 Kapitola 2

62 XMPP

63 V následující kapitole jsou, pro usnadnění a jednodušší pochopení, rozebrány základní sta-
64 vební kameny protokolu Extensible Messaging and Presence Protocol (XMPP). Konkrétně
65 jsou zde popsány stávající vlastnosti implementace, architektura protokolu XMPP obecně
66 [23, 24] a další detaily protokolu [1, 25, 14]. Vzhledem k požadavkům na dolování v da-
67 tech popsaných v následující kapitole je kladen důraz na vybraná rozšíření [22, 12]. Tato
68 rozšíření tvoří základ pro některé rozšířené statusy, jako je například User Tune [18], User
69 Mood [21], User Location [6] a další. Další informace použité pro popis a pochopení XML
70 jazyka byly čerpány z [10, 9].

71 Vznik samotného protokolu XMPP je datován do roku 2004 (březen), kdy na něj byl
72 přejmenován Jabber. Původní projekt Jabber byl vytvořen roku 1998 autorem Jeremie
73 Millerem, který ho založil za účelem vytvořit svobodnou otevřenou IM službu. Uvedený
74 projekt měl obsahovat tři základní vlastnosti, do kterých se zahrnují jednoduchost a sro-
75 zumitelnost pro implementaci, jednoduchost v oblasti šíření a otevřenost podobě veřejně
76 dostupného popisu samotného protokolu. Základní vlastnosti a výhody klientů a serverů
77 budou podrobněji popsány níže. Roku 1999, 4. ledna byl vytvořen první server se jménem
78 Jabber. Komunita vývojářů se chopila iniciativy a vytvořila klienty, kteří dokázali se ser-
79 verem komunikovat, pro různé platformy (Linux, Macintosh, Windows). Roku 2004 byl
80 protokol XMPP přidán mezi RFC¹ dokumenty. Základní norma popisující obecnou struk-
81 turu protokolu je RFC 3920 [23] a RFC 3921 [24], který se zaměřuje na samotný instant
82 messaging a zobrazení stavu. Další zdokumentovaná rozšíření jsou vydávána v podobě tzv.
83 XEP (XMPP Extension Protocol) dokumentů, které jsou známé také pod starším názvem
84 JEP (Jabber Enhancement Proposal). Dnešní počet těchto norem se blíží k číslu 300. Každý
85 XEP obsahuje stav vývoje (schválení), ve kterém se zrovna nachází.

86 Jako bezpečnostní prvky jsou zde podporovány SASL, TLS a GPG. XMPP protokol
87 je postaven na obecném značkovacím jazyce XML, proto vlastnosti popsané dále v této
88 kapitole platí i pro tento protokol.

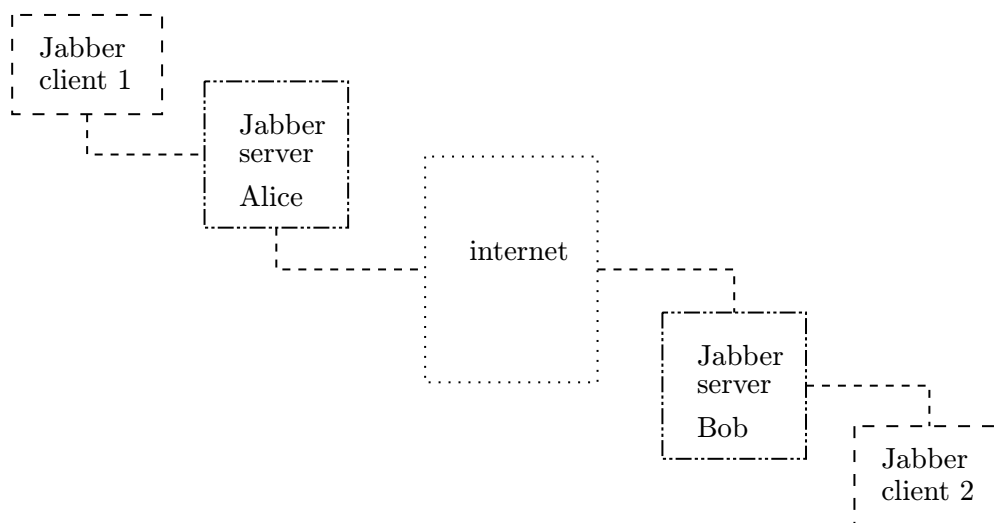
89 2.1 Architektura

90 Dobře navržená internetová technologie je tvořena správně fungujícími komponenty, které
91 mezi sebou dokáží vytvořit spojení a následně započít komunikaci. Pro popis Jabber ar-
92 chitektury v této práci bylo čerpáno z [1, 25]. Tato struktura se nejvíce podobá struktuře
93 posílání e-mailů. Hlavní předností Jabber sítě je, tak jako u elektronické pošty, její decentra-

¹RFC request of comments – žádost o komentáře

lize. V případě Jabberu je decentralizace chápána jako možnost provozovat vlastní server, na rozdíl od jiných komunikačních systémů jako je například Facebook, kde existuje pouze jediný poskytovatel služby. V případě serveru je kladen důraz na spolehlivost a rozšiřitelnost a u klienta na uživatele. Každý server pracuje samostatně, což znamená, že chod ani výpadek jiné datové stanice žádným způsobem jeho běh neovlivní. V případě výpadku jiného serveru bude nedostupný pouze seznam kontaktů a služeb, které registrovaným uživatelům poskytoval.

Obrázek 2.1 znázorňující distribuovanou architekturu Jabberu byl převzat z [1] a doplněn o názvy jednotlivých komponent. Komunikace dvou Jabber klientů probíhá za účasti jejich serverů a sítě, která je spojuje. Spojení mezi nimi bývá často šifrováno.



Obrázek 2.1: Distribuovaná architektura Jabber.

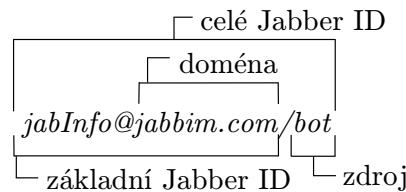
Architektura Jabber serverů využívá velké množství mezi-doménových připojení podobně jako internetový systém elektronické pošty. Komunikace klienta z jedné domény s klientem z jiné na rozdíl od e-mailového modelu nevyžaduje spolupráci třetích stran. Klient se spojí s „domácím“ serverem, který přímo naváže spojení se serverem požadovaného klienta. Tyto vlastnosti jsou zárukou pro bezpečný přenos zpráv, znemožňující „krádeže“ JID², který je popsán níže, a spamování.

Jabber ID

Jabber ID (JID) je jednoznačný virtuální identifikátor uživatele na síti. V případě založení účtu nejsou rozlišována velká a malá písmena, což znamená, že Jabber není case-sensitive. Jednoznačný Jabber identifikátor je složen ze dvou částí: *Jabber bare* neboli čisté ID a *resource* [23]. Základní část na první pohled připomíná e-mailovou adresu *user@server*. Druhá část slouží k přesné identifikaci jednotlivých spojení. Je použita ke směrování síťového provozu s uživateli v případě otevření většího množství spojení pod jedním uživatelem. Společně Jabber bare a resource tvoří tzv. *full JID* — *user@server/resource* například *jabInfo@jabim.cz/bot*. Jednotlivé části uživatelského jména popsané v tomto odstavci jsou ukázány v obrázku 2.2.

²uživatelské jméno

120 Další vymoženost JabberID oproti e-mailové adrese je jeho možnost používat prakticky libovolné národní znaky u doménových jmen a uživatelských účtů [25]. Využíváním kó-



Obrázek 2.2: Rozebraná struktura Jabber ID.

121
122 dování UNICODE, se XMPP stává plně mezinárodní a není jako jiné protokoly omezen
123 rozsahem ASCII tabulky. Přestože je tato vymoženost k dispozici, doposud není žádným
124 výrazným způsobem využívána.

125 Klient

126 Klient je často jednoduchá aplikace pracující se vzdálenými službami, které jsou provo-
127 vány serverem. V této práci je zastoupen robotem s konzolovým rozhraním. XMPP svou
128 architekturou nutí, aby byl co nejjednodušší. Vlastnosti, které by měl mít, jsou shrnuty,
129 podle [14] do tří bodů:

- 130 1. komunikace s jedním Jabber serverem pomocí TCP socketu, který garantuje spolehlivé
131 doručení zpráv na rozdíl od UDP. Nad tímto transportním protokolem dále běží
132 kryptografický protokol TLS, který zabezpečuje komunikaci klient-server a server-
133 server.
- 134 2. rozparsování a následná interpretace příchozí XML zprávy „stanza“ (kapitola 2.3)
- 135 3. porozumění sadě zpráv (*message*, *iq*, *presence*) z Jabber jádra [23]

136 Server

137 Informace použité pro popis XMPP serveru byly čerpány z [14]. K hlavním charakteristi-
138 kám serveru oproti klientovi, jehož základní vlastností byla jednoduchost, patří stabilita a
139 bezpečnost. Je pro něj vyhrazen TCP port 5222. Komunikace mezi servery je realizována
140 přes port 5269. Každý server uchovává seznam zaregistrovaných uživatelů, který nevykazuje
141 žádný jiný server. Zaregistrovaní uživatelé v daném seznamu se mohou do sítě připojovat
142 pouze přes něj. To zajišťuje nemožnost „krádeže“ účtu. Protože XMPP komunikace probíhá
143 přes síť, musí mít každá entita adresu, v tomto případě nazvána JabberID. XMPP spoléhá
144 na DNS což znamená, že používá jména na rozdíl od IP protokolu.

145 Server Jabber je systém spravující tok dat mezi jednotlivými komponentami, které spo-
146 lečně tvoří Jabber služby. Například *Jabber Session Manager* (JSM) poskytne funkce pro
147 IM komunikaci a práci se seznamem kontaktů. Komunikace mezi jednotlivými servery, jak
148 je uvedeno na obrázku 2.1, je zprostředkována za pomoci komponenty *S2S* (server to ser-
149 ver). Při připojení klienta k serveru je komunikace řízená pomocí *C2S* (client to server).
150 Jak již bylo řečeno, Jabber síť využívá doménová jména místo špatně zapamatovatelných
151 IP adres. Pro tento způsob identifikace je určena služba *dnsrv*, která se stará o překlad
152 názvů. V podstatě je to komponenta, která zajišťuje směrování paketů na jiný server.

153 V tabulce 2.1 jsou shrnuty informace o serverech Jabberu. První sloupec tvoří jméno,
 154 následuje programovací jazyk, v němž je napsán. Většina aplikací pro servery je vydá-
 155 vána pod licencí GPL³. U všech aplikací byla zkoumána nejaktuálnější verze. Její číslo
 156 lze nalézt ve třetím sloupci. Všechny servery lze provozovat na operačním systému Li-
 157 nux a Windows. Na platformě Mac OS mohou být použity všechny zde jmenované vyjma
 158 jabberd2. Pět z šesti zde představených programů pro server Jabber jsou stále vyvíjeny,
 159 tedy kromě jabberd14. Hlavním účelem tabulky je prezentovat důležité vlastnosti serverů
 160 v oblasti podpory rozšířených statusů. Jedná se o standardy *pubsub*⁴ (XEP-0060) [12] a o
 161 jeho verzi zaměřenější více na uživatele *pep*⁵ (XEP-0163) [22]. Obě tato rozšíření tvoří ne-
 162 zbytnou základnu pro *rozšířené statusy* a proto je jejich podpora jak u serverů, tak klientů
 163 vyžadována. Podrobněji toto téma bude rozebráno v některé následující podkapitole.

Server	Jazyk	Verze	XEP-0060	XEP-0163
ejabberd	Erlang/ Top	2.1.6	ANO	ANO
Openfire	java	3.6.4	ANO	ANO
jabbred2	c	2.2.11	NE	NE
jabbred14	c, c++	1.6.1.1	ANO	NE
Prosody	lua	0.7.0	NE	ANO
Tigase	java	5.0.0	ANO	ANO

Tabulka 2.1: Přehled Jabber serverů.

164 Z výše uvedené tabulky je zřejmé, že aplikace pro servery, které jsou stále ve vývoji,
 165 podporují tzv. *rozšířené statusy*. Tedy kromě programu jabbred2.

166 2.2 XML

167 Jazyk XML (eXtensible Markup Language) [9], metajazyk pro deklaraci strukturovaných
 168 dat, je jádrem protokolu XMPP. Samotný jazyk vznikl rozšířením metajazyka SGML, jež
 169 slouží pro deklaraci různých typů dokumentů. Základní vlastností je jednoduchá definice
 170 vlastních značek (tagů). Dokument XML se skládá z elementů, které můžeme navzájem
 171 zanořovat. Vyznačujeme je pomocí značek — počáteční a ukončovací. Pomocí tohoto jazyka
 172 je tvořena *stanza* popsána v následující kapitole.

173 Ukázka možné struktury dokumentu psaného jazykem XML je zobrazena na příkladu
 174 2.1. Standardně je předpokládáno, že je psán v kódování UTF-8 [10], ale je-li jako v tomto
 175 případě použito jiné, musí být konkrétní kódování uvedeno na jeho počátku. V opačném
 176 případě nemusí být obsah správně zobrazen. Na začátku dokumentu se také uvádí verze
 177 XML, ve které je dokument psán (1. řádek příkladu). Následuje kořenový element, který
 178 je uzavřen na samotném konci dokumentu. 4. řádek prezentuje možnost použití prázdného
 179 elementu, který obsahuje jeden atribut s názvem zkratky fakulty. Velký význam zde mají
 180 úhlové závorky. Jsou jimi z obou stran obaleny všechny elementy.

³General Public License — všeobecná veřejná licence GNU

⁴Publish-Subscribe

⁵Personal Eventing Protocol

```

1      <?xml version="1.0" encoding="iso-8859-2"?>
2      <fakulta>
3          <název>Fakulta informačních technologií</název>
4          <zkratka fakulty="FIT"/>
5          <typy studia>
6              <bakalářské titul="Bc."></bakalářské>
7              <magisterské></magisterské>
8              <doktorské></doktorské>
9          </typy studia>
10     </fakulta>

```

Příklad 2.1: Ukázka základního XML dokumentu.

2.3 Stanza

Základní jednotkou pro komunikaci založenou na XML je stanza. Z jednoduššího pohledu je možné se na ni dívat jako na jeden dlouhý XML soubor. Při zahájení komunikace se tento soubor „otevře“. Jeho samotné uzavření probíhá až při odhlášení od sítě, neboli přepnutí klienta do stavu offline. Stanzu je tedy možné vnímat jako stream, který obsahuje všechna data probíhající komunikace. Mezi elementy používané pro komunikaci klienta se serverem patří tyto tři: *message*, *presence* a *iq*. Každý zde uvedený člen má svůj jednoznačný význam. V následujících odstavcích jsou jednotlivé části stanzy blíže definovány a na reprezentativních příkladech jsou ukázány jejich základní struktury a možnosti využití v praxi.

První prvek, který bude charakterizován je označen anglickým výrazem *message* (zpráva). Jak již název napovídá, slouží k posílání zpráv všeho druhu. Je to základní metoda pro rychlý přenos informací z místa na místo. Zprávy jsou typu „push“, což znamená, že jsou odeslány a není očekávána žádná aktivita od příjemce, která by přijetí potvrdila. Jedno z dosavadních využití se nachází v klasické komunikace po internetu, tzv. instant messaging (IM). K dalším možným použitím patří skupinový chat a oznamovací nebo upozorňující zprávy. Každá z těchto zpráv je tvořena z minimální povinné struktury. Tak jako u klasické poštovní korespondence nesmí chybět adresa odesílatele a adresa příjemce, kterému je zpráva adresována. Podle možnosti použití jsou zprávy děleny do kategorií. Jmenovitě toto rozdělení implementuje atribut *type*, který může nabývat jednu ze čtyř hodnot. Jsou rozlišovány zprávy pro komunikaci mezi dvěma entitami, skupinový chat, upozornění, chybová zpráva a v neposlední řadě zpráva bez kontextu vyžadující odpověď příjemce. Nakonec nesmí být opomenut blok zprávy, pro uživatele IM nejdůležitější, nesoucí vlastní obsah.

Základní použití struktury elementu *message* je prezentováno na příkladu 2.2. Na prvním řádku je uveden atribut, značící odesílatele. Druhý řádek obsahuje JID klienta, který zprávy přijímá. Následuje informace o typu zprávy a poté je uveden element *body* nesoucí samotný obsah.

```

1      <message from="user@jabber.com"
2              to="jabinfo@jabber.com/bot"
3              type="chat"
4      <body> Kolik je hodin? </body>
5      </message>

```

Příklad 2.2: Použití elementu *message*.

Další částí stanzy je poskytována struktura pro *request-response* (žádost–odpověď) vazbu, podobnou metodám GET, POST a PUT z protokolu HTTP [25]. Zkráceně je ozna-

209 čována pomocí dvou počátečních písmen *Info/Query* neboli *IQ*. Na rozdíl od elementu
 210 *message* tvoří *iq* spolehlivější přenos, optimalizovaný pro výměnu dat (binární data). K
 211 dalším rozdílům patří povinnost příjemce odpovědět na každou přijatou zprávu, neboli po-
 212 tvrdit její doručení. Skutečnost, že je na právě požadovanou zprávu odpovězeno, zajišťuje
 213 parametr *id*. *Iq* dotaz nebo odpověď musí obsahovat stejnou hodnotu tohoto atributu jako
 214 zpráva vytvořená žádajícím subjektem. Další povinný atribut rozděluje *iq* na čtyři typy.
 215 Jednotlivé žádosti na proces nebo akci jsou posílány samostatně [24]. V příloze E je uve-
 216 dena rozsáhlejší struktura tohoto elementu. Použití nachází v případech, které nastavují,
 217 žádají nebo informace posílají. Tato struktura je využívána pro novou registraci, posílání
 218 seznamu kontaktů a další.

219 Příklad 2.3 znázorňuje základní použití elementu *iq*. Uživatel *user* posílá dotaz na získání
 220 seznamu kontaktu (řádek 5.).

```

1      <iq from="user@jabbim.com/doma"
2          to="user@jabbim.com"
3          id="uhhfw23648"
4          type="get"
5      <query xmlns="jabber:iq:roster" />
6      </iq>

```

Příklad 2.3: Použití elementu *iq*.

221 Poslední a pro tuto práci nejdůležitější prvek stanzy je *presence*. V případě, že nemá
 222 určeného příjemce, tak funguje způsobem jako broadcast. Což znamená, že jsou informace
 223 směrovány všem klientům, kteří jsou zaregistrováni k jejímu odběru. Presence v českém
 224 překladu informace o stavu (přítomnost) rozesílá dostupnost ostatních entit v síti. Jedná
 225 se tedy o nastavení uživatelské dostupnosti tak jako na jiných real-time komunikačních a
 226 sociálních systémech.

227 Existuje několik základních stavů statusů, které reprezentují aktuální dosažitelnost uží-
 228 vatele. Tento jev je vyjádřen pomocí elementu *show*, který disponuje čtyřmi možnostmi.
 229 První oznamuje, že je uživatel k dispozici a schopen aktivní komunikace. Druhá často se
 230 vyskytující možnost naznačuje, že je subjekt krátkou dobu pryč od svého IM klienta. Tento
 231 a další dva stavy, popsané dále, jsou často změněny bez lidského zásahu (pomocí pc nebo
 232 jiného zařízení) prostřednictvím funkce známé jako „auto-away“. Poslední dva stavy cha-
 233 rakterizují delší časové období nečinnosti. Tato oznámení o změně stavu uživatele jsou často
 234 zasílána pouze kontaktům, které se nacházejí v režimu online. Tato optimalizace přispívá
 235 ke snížení síťového provozu, jelikož presence v reálném čase při komunikaci využívá velké
 236 množství šířky pásma.

237 Základní použití *presence* je zobrazeno v příkladu 2.4. Kontakt *jabinfo@jabbim.com/bot*
 (1. řádek) posílá informace o svém stavu (řádek č. 2) a svůj status (č. 3).

```

1      <presence from="jabinfo@jabbim.com/bot"
2          <show> online </show>
3          <status> Jsme zde. </status>
4      </presence>

```

Příklad 2.4: Použití elementu *presence*.

238
 239 Obsáhlejší struktura elementu *presence* je zobrazena v příloze E, kde je rovněž k nalezení
 240 přehled všech možných stavů.

241 Jak již bylo zmíněno v části o Jabber ID, Jabber podporuje práci s více současně připoje-
242 nými klienty k jednomu Jabber účtu. Vysvětlení funkčnosti bude prezentováno na příkladu
243 uživatele přihlášeného na stolním počítači a z klienta v mobilním telefonu. U obou těchto
244 připojení je použit stejný Jabber bare, ale odlišného resource, například *domov* a *mobile*.
245 Právě tento rozdíl v tzv. „full“ adrese účtu zajišťuje jednu ze dvou možných podmínek
246 pro správnou adresaci zpráv. Druhá možnost, která bude uplatněna při použití adresy účty
247 pouze ve formě Jabber bare, je nastavení priority u jednotlivých programů. Priorita je číslo
248 v rozsahu hodnot od -128 do 127, kde klient s větší prioritou má přednost před klientem s
249 nižší. Nastane-li případ připojení více klientů se stejnou prioritou, každý server se při roze-
250 sílání zpráv zachová podle vlastní implementace. Některé rozešlou zprávy všem klientům,
251 jiné naopak jen poslednímu přihlášenému.

252 2.4 Rozšíření

253 Dále se tato práce zabývá rozšířeními protokolu XMPP o další vlastnosti, k jejichž popisu
254 slouží XEP. Pro tuto práci jsou nepostradatelné „statusy“, pro které tvoří základ standardy
255 XEP-0060 [12] a XEP-0163 [22] zkráceně PEP⁶. Obě tato rozšíření umožňují strukturovaně
256 pracovat, používat a přenášet další XEP protokoly. Jako příklady relevantní k práci jsou zde
257 uvedeny protokoly *User Location* (kde se uživatel právě nachází) [6], *User Tune* (co uživatel
258 poslouchá za hudbu) [18], *User Mood* (aktuální nálada uživatele) [21] a *User Activity* (co
259 uživatel právě dělá) [11]. Jsou to tedy protokoly založené na PEP, které vyžadují podporu
260 nejen v klientech, ale i na straně serveru (zobrazuje tabulka 2.1). S touto informací úzce
261 souvisí další protokol XEP-0115 [7], který umožňuje zjistit podporované schopnosti klienta,
262 případně, které informace je ochoten přijímat. Tato vlastnost bude popsána níže v části
263 zabývající se podporovanými vlastnostmi.

264 Všechna tato rozšíření by mohla být přidána přímo do statusu viz příklad 2.4, avšak
265 ten je primárně určen k informování o přítomnosti na IM síti. Hlavní rozdíl mezi PEP a
266 obyčejným posílání stavu pomocí presence je v pravomoci klienta přijmout nebo odmítnout
267 informaci, na rozdíl od presence, jež je přijata vždy.

268 Základ přenosu informací začíná na straně klienta, který chce všechny ve svém roster
269 listu (seznam kontaktů), informovat o statusu. Zašle zprávu obalenou v elementu *iq* serveru.
270 Ukázka této zprávy je prezentována na příkladu 2.5, který znázorňuje zaslání informace o
271 druhu hudby, kterou v danou chvíli uživatel poslouchá. Využívá k tomu rozšíření *User*
272 *Tune*, definovaném na řádku číslo 5. Základ zprávy oznamující začátek vysílání informací
273 o rozšířených statusech je vždy stejný. Liší se pouze řádkem 3. a obsahem elementu *item* v
274 příkladu 2.5.

```
1      <iq from='user@jabbim.com' type='set' id='pub1'>
2          <pubsub xmlns='http://jabber.org/protocol/pubsub'>
3              <publish node='http://jabber.org/protocol/tune'>
4                  <item>
5                      <tune xmlns='http://jabber.org/protocol/tune'>
6                          <artist>Daniel Landa</artist>
7                          <length>255</length>
8                      ...
```

Příklad 2.5: Začátku vysílání rozšířeného statusu.

⁶Personal Eventing via Pubsub

275 V případě úspěšného přijetí *iq* zprávy serverem, každý, kdo se zaregistroval k odebrání
276 rozšířených statusů, obdrží oznámení ve formě *message*. Oznámení bude také doručeno všem
277 resources. Celá zpráva i všechny další náležitosti jsou uvedeny v příloze E.

278 Podporované vlastnosti

279 Jednotlivá rozšíření protokolu XMPP jsou nepovinná, a proto nemusí být ve všech klient-
280 ských aplikacích podporována. Pro zjištění podporovaných rozšíření se používá XEP-0115
281 Entity Capabilities [7]. Toto rozšíření výrazně snižuje počet a velikost komunikací a přenosů
282 zpráv mezi uživateli. Dotazem zobrazeným na příkladu 2.6 je zjištěna schopnost jednotli-
283 vých klientů, kterou následně server využije pro správné směrování rozšířených statusů.
284 Všechny zde zmiňované rozšíření a protokoly z této kapitoly je možné u každého klienta
285 (seznam klientů obsahuje tabulka v příloze E) vyčíst z atributu *ver* (druhá část u atributu
286 node), který je vypočítán ze všech podporovaných protokolů klienta, viz [7].

```
1<iq from="user@jabbim.com" id="disco1"  
2  to="jabinfo@jabbim.com/bot" type="get">  
3  <query xmlns="http://jabber.org/protocol/disco#info"  
4    node="http://code.google.com/p/exodus#QgayPKawpkPSDYmwT/WM94uAlu0=" />  
5</iq>
```

Příklad 2.6: Dotaz na podporované protokoly.

287 Další rozšíření

288 V následujících několika odstavcích budou přiblíženy specifikace jednotlivých rozšíření XEP,
289 které slouží jako zdrojová data pro dolování a jsou relevantní k tématu práce.

290 Prvním rozšířením, nad rámec základních vlastností Jabberu, které zde bude podrobněji
291 rozebráno, je elektronická verze klasické vizitky neboli *VCard*. Jeho specifikací se zabývají
292 dva standardy. Jelikož novější verze XEP dokumentu [13] se v době psaní této práce na-
293 cházela ve stavu „experimental“, což znamená, že ještě není schválena jako standard, je
294 pouze ve stavu návrhu. Proto bylo použito verze starší [16]. Jednoduše řečeno je VCard
295 struktura, která nese informace o uživateli jako je jméno, příjmení, e-mail, adresa bydliště
296 i zaměstnání a další údaje. Data jsou dále zveřejňována na síti, z čehož vyplývá, že jsou
297 dostupná ostatním uživatelům. Vyplnění těchto osobních údajů je dobrovolné a tak se u
298 některých uživatelů nachází pouze přezdívka a JID, které jsou často předdefinovány auto-
299 maticky. Nedílnou součástí všech sociálních a komunikačních systému jsou malé fotografie,
300 loga nebo ikony, kterými se uživatelé prezentují. V síti Jabber tomu není jinak, a proto je
301 samotný obrázek zahrnut přímo do VCard v položce *photo*. Podrobnější informace o jeho
302 nastavení a přijímání je možné nalézt v *vCard-Based Avatars* [19], který jej definuje.

303 Díky základní podmínce XMPP protokolu (otevřenost) existuje mnoho různých aplikací,
304 pomocí kterých lze v síti Jabber komunikovat. S programy, používanými uživateli, úzce
305 souvisí další zde implementované rozšíření. Jedná se o realizaci *Software Version* dokumentu
306 [17], který se právě zabývá získáváním informací o samotných aplikacích. Je-li toto rozšíření
307 podporováno je díky němu možné zjistit jméno a verzi používané aplikace. Informace o
308 operačním systému často nejsou kvůli bezpečnosti ani vyplněny. Podrobnější informace o
309 softwarové výbavě klienta je možné zjistit pomocí XEP [7], o kterém již bylo dříve psáno v
310 odstavci zabývajícím se podporovanými vlastnostmi klientských aplikací.

311 S rozšířením tzv. „chytrých“ mobilních zařízení mezi širší veřejnost vzniklo několik no-
312 vých disciplín spojených s určováním zeměpisné polohy, jako je například geocaching. Geo-
313 grafická poloha je přenášena ve formě souřadnic popisující přímo zeměpisnou šířku a délku.
314 Současně lze informaci o poloze přenášet i slovně ve formě adresy. Příkladem slovního po-
315 pisu je ulice, číslo popisné, město a další. Mnoho aplikací, které mají k dispozici GPS
316 přijímač, vysílají a aktualizují zeměpisné informace automaticky, například po určité době
317 nebo změně polohy o určitou vzdálenost. Toto a další níže popsání rozšíření jsou postaveny
318 na již zmiňovaném PEP. Některé části protokolů jsou zjednodušeny a připraveny tím pro
319 „mobilní instant messaging“.

320 Pro sdělení informací o stavu klienta není v základní verzi Jabberu mnoho. Pomocí
321 presence je možné „pouze“ prozradit, zda je uživatel připraven komunikovat nebo je mo-
322 mentálně nedostupný a to v několika verzích lišících se délkou nepřítomnosti. Pokročilejší
323 nastavení statusu nabízí *User Mood* [21] a to ve formě sdělení současné nálady, jako je
324 například radost. Další možné upřesnění činnosti uživatele jsou definovány v *User Activity*
325 [11], kde každá činnost je složena z povinné obecné kategorie a nepovinné, která informaci
326 upřesňuje. Příkladem může být *eating* a *having_a_snack* tj. uživatel jí, uživatel svačí.

327 K poslednímu rozšíření implementovanému v této práci patří *User Tune* [18], které
328 umožňuje uživateli šířit informace o aktuálně poslouchané hudbě. Některé dnešních hu-
329 dební přehrávače dokáží automaticky spolupracovat s IM klientem a předávat informace o
330 hudbě bez nutného lidského zásahu. Ve zprávě jsou tedy přenášeny informace o skladbě,
331 interpretovi, albu a další informace, které mohou být získávány z MP3 ID3v1 nebo novější
332 ID3v2 tag.

333 Podpora výše popsaných rozšíření v aplikacích je poměrně malá. Například v předchá-
334 zející zmiňované části o poslouchané hudbě, při stavu, kdy program toto rozšíření nepod-
335 poruje, je posíláno pomocí normální presence. Jméno skladatele, alba a další podrobnosti
336 jsou shrnuty do statusu, tudíž jsou doručeny všem uživatelům ze seznamu kontaktů.

337 Kapitola 3

338 Data mining

339 Třetí kapitola se zabývá procesem dobývání znalostí z databází. Popisuje jej jako disciplínu,
340 která vznikla za účelem vytěžení informací z dat, která jsou v nepřehledném množství uklá-
341 dána v databázích. Díky velikosti dnešních disků, objem ukládaných dat neustále roste. S
342 tím také úzce souvisí zvětšující se poměr nepotřebných a zašumělých dat vůči užitečným
343 informacím. V této kapitole jsou mimo jiné popsány metody používané k dolování z dat,
344 které jsou relevantní k této práci.

345 Na začátku kapitoly je rozebrán pojem získávání znalostí databází, jehož jednu pod-
346 statnou část tvoří samotný data mining. Dále je vysvětlena základní terminologie, pro kterou
347 bylo čerpáno z [8]. Cílem první podkapitoly je přiblížení způsobu, jakým byla data uložena
348 v databázi, připravena k samotnému data miningu. Celá druhá podkapitola je věnována vy-
349 braným metodám pro dolování dat a vlastnostem, které je od sebe navzájem odlišují. Jsou
350 zde rozebrány *asociační pravidla*, pro jejichž popis bylo čerpáno z [2]. Pro ostatní metody,
351 které jsou popsány dále, byla jako zdroj informací použita kniha [5]. Poté následuje třetí
352 podkapitola, která se podrobněji zabývá jednou z metod pro dolování dat a to *shlukováním*.
353 Obsahem této části jsou již konkrétní algoritmy pro shlukování dat [28, 3] a také metoda
354 *k-Means* využívaná v praktické části této práce. Kapitulu uzavírá stručný přehled vybra-
355 ných programů pro data mining a podrobnější seznámení s nástrojem *RapidMiner*, který je
356 v této práci využíván pro samotné dolování.

357 Terminologie

358 Pojem data mining neboli česky dolování dat se začal ve vědeckých kruzích objevovat počát-
359 kem 90. let 20. století. První zmínka pochází z konferencí věnovaných umělé inteligenci (IJ-
360 CAI'89¹—mezinárodní konference konaná v Detroitu, AAAI'91² a AAAI'93—americké kon-
361 ference v Californii a Washingtonu, D.C) [2].

362 Tradiční metoda získání informací z dat je realizována jejich manuální analýzou a inter-
363 pretací. V praxi ji například nalezneme v odvětví zdravotnictví, vědy, marketingu (efektivita
364 reklamních kampaní, segmentace zákazníků) a dalších. Pro tyto a mnoho dalších disciplín
365 je manuální zpracování příliš pomalé, drahé a vysoce subjektivní. Další důvod k přechodu
366 na jiné metody je objemnost dat, která dramaticky vzrostla a tudíž se manuální analýza
367 stává zcela nepraktická. Databáze rychle rostou ve dvou následujících kategoriích:

- 368 1. počet záznamů neboli objektů v databázi

¹International Joint Conference on Artificial Intelligence

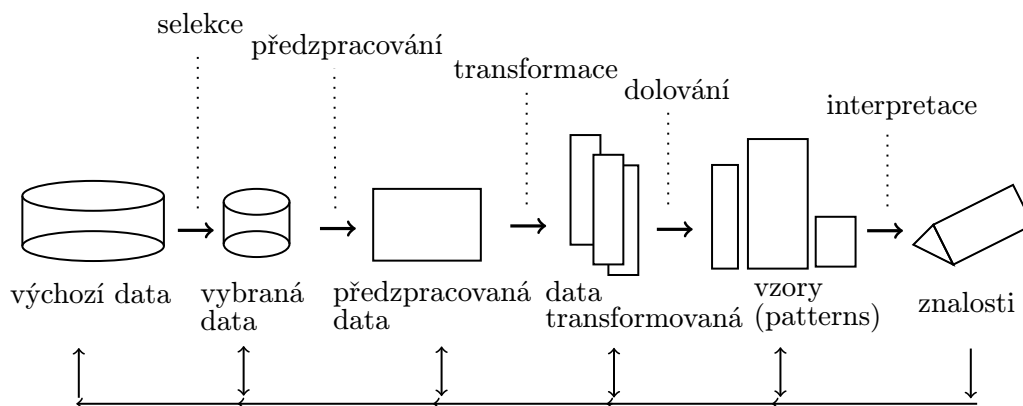
²Association for the Advancement of Artificial Intelligence

369 2. počet polí neboli atributů objektů v databázi

370 Proces data mining je pouze jedna část z odvětví nazývané dobývání znalostí z da-
371 tabází neboli KDD³ definované níže v definici 3.0.1. Vznik disciplíny KDD je důsledkem
372 nepřehledného množství automaticky sbíraných dat, která je potřeba dále využívat. Pod-
373 statným znakem celého procesu je správnost reprezentace výsledků formou, která má k
374 uživateli nejbližší. Jako příklad bude uvedena implikace ve tvaru rozhodovacích pravidel,
375 asociační pravidla, rozhodovací stromy, shluky podobných dat a další. Základem KDD je
376 praktická použitelnost metod. Očekává se zjištění nových skutečností namísto prezentování
377 již známých informací.

378 **Definice 3.0.1** KDD je chápáno jako interaktivní a iterativní proces tvořený kroky se-
379 lekce, předzpracováním, transformace, vlastního „dolování“ (data-mining viz 3.0.2) a in-
380 terpretace [2].

381 Grafické znázornění definice 3.0.1 je popsáno schématem na obrázku 3.1, který pre-
382 zentuje časový harmonogram v KDD. Schéma znázorňuje následnost jednotlivých procesů,
383 které tvoří KDD. KDD je iterativní proces, z čehož vyplývá, že skutečnosti nalezené v pře-
384 dešlých částí zjednoduší a zpřesní vstupy pro následující fáze. Jakmile jsou znalosti získány,
385 jsou prezentovány uživateli. Pro přesnost může být část procesu KDD ještě upravena. Tím
386 budou získány „přesnější a vhodnější“ výsledky.



Obrázek 3.1: Proces dobývání znalostí z databází podle knihy autora Fayyad [4].

387 Vzhledem k obrázku 3.1, který prezentuje jednotlivé kroky získávání znalostí z databází
388 budou dále tyto procesy popsány. Prvním část v KDD je tvořena výchozími daty, které slouží
389 jako zdroj pro ostatní fáze. Samotný popis získávání těchto dat je popsán v předcházející
390 kapitole. Procesu selekce dat, je kladen za cíl, vybrat co možná „nejúčinnější“ množinu
391 dat a tím i zmenšit její celkový objem. V této části získávání znalostí se při vybírání dat
392 bere ohled na to, jak se jednotlivá data vztahují ke konkrétnímu uživateli. Následující proces
393 nazvaný transformace se zabývá převedením dat do vhodného formátu pro samotné dolování
394 informací. Tato část je popsána v následující podkapitole, kde hlavní úlohu při transformaci
395 je čas. Vybrané metody pro dolování dat jako je například shlukování a další, jsou taktéž
396 v této práci popsány níže.

³Knowledge Discovery in Database

397 Získávání znalostí z databází je proces složený z několika kroků vedoucích od surových
398 dat k formě nových poznatků. Iterativní proces je složený, tak jak je prezentováno v [5], z
399 následujících kroků:

- 400 • **čištění dat** – fáze, ve které jsou nepodstatné údaje odstraněny z kolekce.
- 401 • **integrace dat** – kombinování heterogenních dat z několika zdrojů do společného
402 jediného zdroje.
- 403 • **výběr dat** – rozhodování o relevantních datech.
- 404 • **transformace dat** – také známý jako konsolidace dat. Fáze, ve které jsou vybraná
405 data transformována do formy vhodné pro dolování.
- 406 • **data mining** – zásadní krok, ve kterém jsou aplikovány vzory na data.
- 407 • **hodnocení modelů** – vzory dat zastupují získané znalosti.
- 408 • **prezentace znalostí** – konečná fáze, zjištěné poznatky jsou reprezentovány uživa-
409 teli. Tento základní krok využívá vizualizační techniky, které pomáhají uživa-
410 telům porozumět a správně interpretovat získané výsledky.

411 Jak je uvedeno v [5], běžně jsou některé z těchto kroků kombinovány dohromady. Kroky
412 čištění dat a integrace dat mohou být provedeny společně, tak jako to prezentuje schéma
413 na obrázku 3.1.

414 V této podsekcí jsou ve stručnosti vysvětleny základní nejdůležitější pojmy dále v práci
415 využívané.

416 Definic výrazu data mining se v odborné literatuře nachází několik. Zde uvedená je
417 kombinací dvou „definic“ z [15].

418 **Definice 3.0.2** Data Mining je proces objevování znalostí, který používá různé analytické
419 nástroje sloužící k odhalení dříve neznámých vztahů a informací z velmi rozsáhlých databází.
420 Výsledkem je predikční model, který je podkladem pro rozhodování [15].

421 Mezi další čteně se vyskytující pojmy v tomto odvětví patří například data, znalosti a
422 informace. Tyto termíny jsou často mezi sebou zaměňovány, proto jsou níže jejich významy
423 striktně definovány tak jako v [8].

424 Jedna z několika existujících definic pojmu data je uvedena v definici 3.0.3, která je po-
425 pisuje z pohledu informačního. Data často nemají sémantiku (význam) a bývají zpracována
426 čistě formálně.

427 **Definice 3.0.3** Data jsou z hlediska počítačového pouze hodnoty různých datových typů.

428 Informace lze chápat jako data, která byla obohacena o sémantiku (význam), jsou tedy
429 již zpracovaná a interpretována uživatelem. Znalosti, jsou řazeny do stejné kategorie jako
430 informace, ale jejich interpretace bývá ještě složitější. Často bývají tvořeny shluky informací,
431 proto jsou reprezentovány jako odvozené informace. Podle studijní opory [8] jsou znalosti
432 informace, které jsou zařazeny do souvislostí.

433 3.1 Transformace dat

434 Transformace dat tvoří třetí část z celkového procesu dobývání znalostí z databází. Než se
 435 data dostala do tohoto stavu, bylo na nich provedeno několik kroků, ve kterých byla upra-
 436 vována. V první fázi byla sbírána Jabber komunikace, která je popsána v první kapitole.
 437 Druhá fáze byla zaměřena na zúžení výsledné množiny, a proto byla vybrána jen relevantní
 438 data. I přes tyto kroky relační databáze obsahuje velké množství dat, která se nenachází ve
 439 stavu, aby mohla být použita jako zdroj pro data mining. Jak bude popsáno v následující
 440 podkapitole většina metod pro dolování dat pracuje pouze s daty, která obsahují kvanti-
 441 tativní proměnné. Za tímto účelem je potřeba všechny atributy tabulky z databáze, které
 442 mají být nadále používány, převést na měřitelné hodnoty.

V této práci se bude pracovat s atributy nesoucí informace o jak aktuálních tak minulých
 stavech uživatelů, kteří si přidali účet *jabInfo@jabbm.com* do svého seznamu kontaktů. A
 tak byla jejich každá změna statusu uložena do databáze. Z důvodu nečíselného hodnoty
 stavů, jako je například *available*, *away* a další, je třeba provést jejich transformaci na
 kvantitativní hodnoty. Proces byl proveden pomocí bijektivního zobrazení. Kde zobrazení
 je, podle [8], funkce s definičním oborem S a oborem hodnot T , která je nazývána binární
 relace $f \subseteq S \times T$. V této relaci se nevyskytují dvě různé dvojice (s, t_1) a (s, t_2) , kde $s \in S$
 a $t_1, t_2 \in T$. Prvky t_1 a t_2 jsou různé. Z toho vyplývá, že každému prvku s z množiny S je
 přiřazen jednoznačně právě jeden prvek $t \in T$. Tuto definici je možné zapsat ve tvaru:

$$f : S \rightarrow T,$$

443 kde S a T jsou množiny (D_f, H_f) .

Konkrétní případ transformace z této práce tedy bude obsahovat množinu S , kde

$$S = \{Available, Chat, Away, DND, XA, Unavailable\}$$

a množina T , kde

$$T = \{120, 110, 90, 70, 50, 0\},$$

444 do které budou jednotlivé prvky z množiny S bijektivně zobrazeny. Kdy bijekce je zobrazení,
 445 které každému prvku z cílové množiny, konkrétně z množiny T , přiřazuje právě jeden prvek
 446 z množiny počáteční, tedy S .

447 Při výběru velikosti hodnoty, pro výslednou množinu T , bylo čerpáno z programu Ga-
 448 jim, který je multiplatformní klient s velkou podporou standardů a rozšiřujících proto-
 449 kolů. Hodnoty v množině T byly zvoleny tak, aby měly sestupné uspořádání, a aby bylo
 450 možné je dobře mezi sebou porovnávat. Jsou-li vybrány dvě hodnoty například *available* a
 451 *unavailable*, vzdálenost mezi nimi musí být větší než vzdálenost například u hodnot *chat* a
 452 *away*. Na druhou stranu prvky ze vstupní množiny S jsou striktně definovány podle Jabber
 453 standardu, který je popsán v RFC [23].

454 Temporální data

455 Druhá podstatná transformace, pro kterou bylo čerpáno z [26], se tak jako první nezabývá
 456 transformováním dat textových na data, jejichž obsah by byl tvořen kvantitativními pro-
 457 měnnými. V této části jsou řídká temporální data transformována na hustá. Pro následné
 458 vyhodnocení a data mining je potřeba řádkům z tabulky presence přidat konečné časové
 459 razítko, které by vymezilo interval doby platnosti těchto dat.

460 Jak již bylo uvedeno, hlavním rozdílem mezi temporálními databázemi a ostatními je
 461 schopnost uchovávat časové údaje. Své uplatnění nachází v odvětvích, kde je potřeba zpraco-
 462 vávat stará a zároveň nová data, například v oblastech medicíny, finančnictví, monitorování
 463 a dalších. Jednotlivé záznamy, které jsou závislé na čase, jsou v databázích ukládány jako
 464 samotné body, tedy diskrétně. Přestože v reálném světě je většina těchto údajů z pohledu
 465 času spojitých.

466 K dalšímu popisu temporálních databází nyní budou charakterizovány tři důležité po-
 467 jmy, pomocí nichž jsou databáze dále děleny. Prvním pojmem je *granualita*, která udává
 468 nejmenší časovou jednotku, kterou databáze rozlišují. Hodnoty, které může nabývat, jsou
 469 hodina, den, rok a další. Velikost granuality ovlivňuje velikost objemu dat, který je přímo
 470 úměrný s přesností záznamů. Dalším pojmem je *čas platnosti*, která reprezentuje období,
 471 kdy je daný fakt v modelovém světě pravdivý. Posledním termínem je *čas transakce*, která
 472 definuje přítomnost faktu v databázi a možnost jej získat. Čas transakce a čas platnosti jsou
 473 na sobě nezávislé a definují dvě rozdílné časové osy, kdy každá může disponovat s jinou gra-
 474 nualitou. Souhrnný název pro výše uvedené tři pojmy, který se používá, je systém časových
 475 razítek. Při použití těchto systému lze následně tvořit dotazy zaměřené na různá časová
 476 období, jako je minulost, přítomnost a budoucnost. Tyto dotazy jsou velmi jednoduché a to
 477 díky rozšířenému jazyku TSQL, který vychází z klasické podoby dotazovacího jazyka SQL.

478 Temporální databáze jsou rozděleny, podle systému časových razítek, na tyto základní:
 479 *snímková*, *transakční*, *platného času*, *obojího času* (bitemporální). Entity z databáze, která
 480 je použita v této práci, je možné zařadit do kategorie *snímkových tabulek* (snapshot). K
 481 jejím hlavním rysům patří zaznamenávání stavu dat v jistém okamžiku. Čas transakce ani
 482 čas platnosti zde nejsou uplatněny.

483 Konkrétní příklad možné transformace je ukázán na části tabulky *presence* 3.1, která
 484 se ve stejném formátu nachází i v databázi, a modifikované tabulce *presence_modify* 3.2.
 Tabulka 3.2 se od původní liší přidáním sloupce *dateEnd* (podbarven šedě), který s atri-

id	date	toj	presence
87365	2011-4-4 13:53:59	JabInfo@jabbim.cz	Available
87369	2011-4-4 15:43:07	JabInfo@jabbim.cz	Away
...

Tabulka 3.1: Ukázka tabulky *presence*.

485 butem *dateStart* vymezuje interval, kdy daná hodnota byla nebo je platná. Tato entita
 486 je pouze příkladem jak by mohla daná transformace vypadat. V této práci se žádná nová
 487 tabulka nevytvářela a ani se nemodifikovala již vytvořená. Celá transformace je popsána v
 488 další části této podkapitoly.

id	dateStart	dateEnd	toj	presence
87365	2011-4-4 13:53:59	2011-4-4 15:43:07	JabInfo@jabbim.cz	Available
87369	2011-4-4 15:43:07	INF	JabInfo@jabbim.cz	Away
...

Tabulka 3.2: Ukázka modifikované tabulky *presence_modify*.

489 Postup jednotlivých kroků při transformaci časových údajů z tabulky *presence*, je zob-
 490 razen v následujícím výčtu. Tento zjednodušený popis algoritmu vyžaduje dva vstupní
 491

parametry. První je JabberID uživatele, jehož položky v databázi mají být transformovány. Druhá nutná položka je datum, které bude sloužit jako upřesňující vstupní interval.

Data z tabulky presence jsou transformována na vektor ϑ o 288 dimenzích, kde z pohledu časového je jedna dimenze období vymezené 5 minutami. Den je rozdělen na úseky po 5 minutách ($\delta = 300s$), kterých je 288. Tedy

$$\vartheta = (\vartheta_1, \vartheta_0, \dots, \vartheta_N),$$

kde $N = 288$.

1. Vstupním parametrem je datum, které vymezuje data pro transformaci. Toto datum je převedeno na dvě data (počátek a konec dne), která tvoří hraniční body v intervalu ι .
2. Výběr dat z databáze, která splňují časové období definované intervalem ι a uživatel ID . Uložení těchto dat do množiny Γ .
3. Pokud zadanému dotazu neodpovídá žádný řádek z tabulky, jsou data označena jako prázdná. Výstupní transformovaný vektor ϑ je naplněn hodnotami reprezentujícími stav *Unavailable*. Algoritmus je **ukončen**.
4. Zjištění prvního statusu toho dne.
 - 4.1 Převod data o den dřívejšího na interval ι_1 , například $\langle 2011-08-22\ 00:00:00, 2011-08-22\ 23:59:59 \rangle$.
 - 4.2 Výběr dat vyhovujícím intervalu ι_1 a uživateli ID z databáze.
 - 4.3 Výběr posledního záznamu z množiny dat získaných z předešlého dotazu.
 - 4.4 Nalezení poslední presence a uložení její hodnoty do λ .
5. Výběr následujícího záznamu z množiny Γ .
6. Výpočet zda je časový interval mezi vybraným a následujícím záznamem větší jak δ .
 - 6.a Časový interval je větší než interval δ .
 - 6.1 Do výsledného vektoru ϑ je ukládána hodnota presence daného záznamu.
 - 6.2 Opakuj předešlý bod **6.1** kolikrát je interval δ menší než rozdíl mezi časy vybraného a následujícího záznamu.
 - 6.b Časový interval je menší než interval δ .
 - 6.1 Jsou vybírány další záznamy z množiny Γ dokud rozdíl mezi časy v sekundách není větší než interval δ .
 - 6.2 Jednotlivé presence záznamů jsou ukládány do pomocného pole, transformované do kvantitativních hodnot, jak je uvedeno v úvodu této podkapitoly.
 - 6.3 Každý prvek pole je vynásoben počtem sekund, zastoupených v daném intervalu.
 - 6.4 Výběr největšího prvku z pomocného pole a uložení jej do výsledného vektoru ϑ .
7. Celý proces je opakován od bodu **5**, dokud množina Γ není prázdná.

524 3.2 Metody dolování dat

525 Základ metod dolování dat je založen na statistice, posledních poznatcích z umělé inteli-
526 gence či strojového učení. Hlavní cíl těchto netriviálních metod je společný — snaha zjištěné
527 výsledky prezentovat srozumitelnou formou. Pro většinu používaných metod je společná
528 vlastnost předpoklad, že objekty popsané pomocí podobných charakteristik patří do stejné
529 skupiny (učení na základě podobnosti similarity-based learning). Objekty obsahující atri-
530 buty, lze převést na body v n -rozměrném prostoru, kde n reprezentuje počet atributů.
531 Vychází se z představy podobnosti bodů tvořící určité shluky v prostoru.

532 Další rozdíly mezi metodami, které byly prezentovány v [2], spočívají v:

- 533 • schopnosti reprezentace shluků (např. otázka lineární separability)
- 534 • srozumitelnosti nalezených znalostí pro uživatele (symbolické vs. subsymbolické me-
535 tody)
- 536 • efektivnosti znovupoužití nalezených znalostí
- 537 • vhodnosti typů dat
- 538 • a další...

539 Problémy, které data mining řeší, se rozdělují do několika skupin. Do výčtu vybraných z
540 nich, které budou následně rozebrány, patří *asociační pravidla*, *klasifikace*, *modely*, *predikce*
541 *a shlukování*.

542 Při popisu asociačních pravidel, která jsou založena na syntaxi *IF-THEN*, bylo čerpáno
543 z [2]. Jejich rozšíření se datuje do 90. let 20. století, kdy byly panem Agrawalem před-
544 staveny v souvislosti s analýzou „nákupního košíku“. Použitelnost bude vysvětlena právě
545 na příkladu analýzy nákupního košíku. Podstata příkladu je tvořena zákazníkem a jeho
546 systémem nakupování. Jsou zjišťovány produkty, které jsou nakupovány současně. Hledají
547 se neboli jsou vytvářeny společné vazby (asociační pravidla) mezi výrobky a určuje se je-
548 jich spolehlivost. Na základě těchto závislostí je upravováno umístění jednotlivých výrobků.
549 Obecně jsou tedy asociační pravidla považována za konstrukci, která z hodnot jedné trans-
550 akce odvozuje možnost výskytu závislostí v jiných transakcích. Jsou tedy hledány všechny
551 vnitřní závislosti existující mezi daty.

552 K dalším metodám pro data minig patří klasifikace, která bude opět vysvětlena na pří-
553 kladu, převzatého z [8]. Podle obsahu databáze nebo dotazníku bude každý klient banky
554 zařazen do různých krizových skupin. Na základě těchto skupin pracuje „credit skóring“,
555 jež klientovi poskytne nebo odepře například úvěr v bance. Další příklady využití jsou na-
556 příklad ve zdravotnictví. Na základě zdravotního stavu pacienta a jeho příznaků, je pacient
557 zařazen do tříd, které reprezentují jednotlivé nemoci. Klasifikací jsou, podle [5], jednotlivé
558 zkoumané elementy rozděleny (podle hodnot atributů) do vhodných kategorií, které jsou
559 předem vytvořeny z navzájem podobných objektů (tvorba profilů třídy). Při této metodě je
560 upřednostňována přesnost před jednoduchostí a rychlostí. Zdroje klasifikovaných objektů
561 jsou většinou tvořeny jednotlivými řádky v databázi. Vzory dat vytváří instance, jejichž
562 vlastnosti reprezentují atributy vyjádřené číselnou hodnotou.

563 Na modelech je založen třetí typ metod pro dolování dat. Základem modelů jsou tré-
564 novací data. Dále uvedené příklady vybraných klasifikačních modelů, byly čerpány z [5].
565 Především jsou to rozhodovací stromy, neuronové sítě, statistické metody, klasifikační pra-
566 vidla a další.

Metoda, která je postavena na myšlence, kde chronologicky seřazená data a vývoj jejich hodnot v minulosti tvoří základ pro určení hodnot budoucích, se nazývá predikce. Je řazena mezi velmi známé procesy, které na základě získaných znalostí předpovídají následující vývoj. Předpokládá se, že na základě informací získaných z dat v minulosti, bude možné postavit modely, které se budou chovat stejně nebo alespoň podobně i v budoucnu. Využití naleznu v předpovědi počasí (z naměřených meteorologických hodnot se určují budoucí předpokládané teploty), při vývoji cen na burze a dalších. Podklady pro popis predikce byly čerpány z [2, 5].

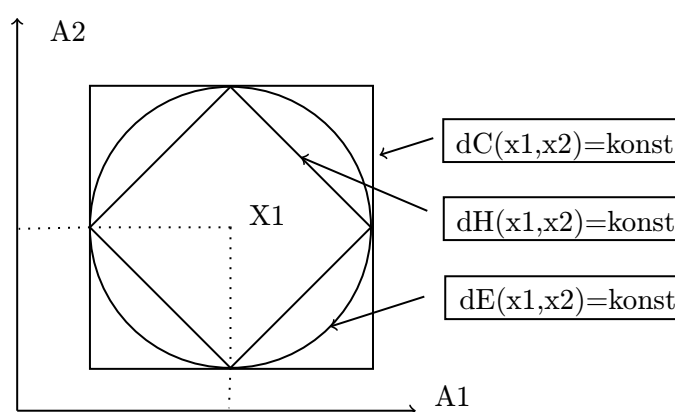
Poslední zde uvedená metoda je zaměřená na dělení objektů do předem neznámých skupin. Proces dělení probíhá na základě specifikace objektů a jejich odlišnosti od ostatních shluků. Tato část, pro kterou bylo čerpáno z [28, 3], bude podrobně rozebrána v následující podkapitole.

3.3 Shlukování

V této podkapitole je shlukování rozděleno na několik metod shlukové analýzy podle [5]. U každé z nich jsou popsány její základní vlastnosti a uvedeny algoritmy relevantní k práci. Poslední metoda *metoda rozkladu* je rozebrána podrobněji z důvodu jejího praktického využití v této práci.

Shlukování je zaměřeno na dělení objektů do předem neznámých skupin. Proces dělení probíhá na základě specifikace objektů a jejich odlišnosti od ostatních shluků.

Většina níže popsaných metod a algoritmů je založena na výpočtu vzdáleností mezi objekty. Tato vzdálenost lze vyjádřit různými mírami, podle knihy [2] například pomocí *Hammingovy vzdálenosti* (dH), *Euklidovské vzdálenosti* (dE) a *Čebyševovy vzdálenosti* (dC). Rozdíl mezi těmito typy určující vzdálenosti, graficky vyjadřuje obrázek 3.2. Kde $X1$ je střed, od něhož jsou jednotlivými obrazy znázorněny dané vzdálenosti. Konkrétně pomyslné body umístěné po obvodu kruhu jsou všechny stejně vzdáleny od středu $X1$. Tato vzdálenost je označena jako Euklidovská. Další 2D těleso čtverec, který je vodorovný s osami $A1$ a $A2$ prezentuje Čebyševovu vzdálenost. Po obvodu posledního obrazce, čtverce otočeného o 45° podle osy $A1$, jsou všechny pomyslné body stejně vzdáleny od bodu $X1$ Hammingovou vzdáleností.



Obrázek 3.2: Srovnání výpočtu vzdáleností od bodu x_1 [2].

Jako první jsou zde rozebrány *metody založené na modelu*, které se pokouší přiřadit

597 data k určitému matematickému modelu na základě společných optimalizovaných vlastností.
598 Většina procesů je založena na předpokladu, že jsou data generována pomocí standardních
599 statistik. Mezi zástupné metody této shlukovací analýzy se řadí Expectation–Maximization
600 (EM) a Self Organizing Oscillator Network, dále jen SOON. Algoritmus SOON je založen na
601 neuronové síti. Je to metoda vycházející z algoritmu SOM⁴ [28]. Metoda EM je rozšířením
602 algoritmu *k-means*, který bude podrobně rozebrán v následující části.

603 Hlavní princip *metody hierarchického shlukování* je založen na tvorbě stromové hierar-
604 chie shluků, která je známá pod názvem *dendrogram*. Hierarchické metody, podle [5], mohou
605 být rozděleny do dvou skupin a to na základě principu, kterým jsou dendrogramy vytvářeny.
606 První možnost je *aglomerativní přístup*, který shlukuje menší shluky, kdy výsledkem je jen
607 jeden. Druhý přístup, *divizní*, je založen na opačném předpokladu. Na počátku je tedy jeden
608 velký shluk, který je postupně rozdělován, dokud není počet shluků roven počtu objektů
609 [28]. Mezi zástupce této metody například patří algoritmus AGNES⁵.

610 K dalším metodám patří *metody založené na mřížce*, které kvantují datový prostor do
611 konečného počtu pravoúhlých buněk. Tyto buňky jsou uspořádány do víceúrovňové mříž-
612 kové struktury. Zmíněná struktura tvoří základ pro shlukové operace. Hlavní výhoda tohoto
613 přístupu je rychlost zpracování, které většinou nebere ohled na počet datových objektů. Čas
614 zpracování závisí pouze na počtu buněk v každé dimenzi kvantovaného prostoru. Mezi zá-
615 stupce metod založených na mřížce patří metoda STING — STatistical INformation Grid,
616 který pracuje se statickými informacemi uloženými v buňkách mřížky. Algoritmus je roz-
617 dělen do dvou částí. První si klade za cíl rekurzivně rozdělit datový prostor na pravoúhlé
618 buňky. Druhá fáze testuje spojitost mezi sousedy relevantních buněk [28]. Mezi další metody
619 založené na mřížce patří WaveCluster⁶, využívající vlnkové transformace k rozdělení pro-
620 storu dat. Tato transformace zdůrazňuje shluky v prostoru a objekty jím vzdálené potlačuje
621 [5].

622 *Metody založené na hustotě* vychází z *m-rozměrného* prostoru, ve kterém jsou zobrazeny
623 objekty ve formě bodů. Místa v prostoru s větší koncentrací objektů ve srovnání s ostatními
624 oblastmi jsou nazývány shluky. Výchozí předpoklad je existence okolí jednotlivých bodů
625 (sousedství). Jedna z charakteristik metod založených na hustotě je schopnost vypořádat
626 se s vzdálenými hodnotami, označovanými jako šum [28]. Jako příklad je uvedena metoda
627 DBSCAN⁷, která je založena na hustotě objektů v prostoru. U jednotlivých objektů je
628 zkoumáno jejich okolí. Algoritmus je ovlivňován dvěma parametry, velikostí shluku ϵ a
629 minimálním počtem objektů v daném shluku *MinPts*, které spolu úzce souvisí (viz [5]).
630 Bod splňující obě podmínky je označen za jádro. Za pomoci jader je rozšiřována množina
631 objektů spojených na základě hustoty. Obsahuje-li jádro x_1 ve svém okolí další jádro x_2
632 znamená to, že jádro x_1 je přímo dosažitelné z jádra x_2 . Tímto způsobem jsou vytvářeny
633 výsledné *shluky*. V opačném případě, body, které nesplňují dvě zmíněné podmínky, jsou
634 označeny jako *šum*.

635 Všechny zde doposud zmiňované metody poskytují dobré výsledky pouze s malým poč-
636 tem dimenzí, tak jak je to popsáno v [8]. S narůstajícím počtem atributů roste počet
637 nerelevantních dimenzí určených pro shlukování. S tímto také přibývá zvětšená produkce
638 zašumění a znesnadnění nalezení relevantních shluků. Data jsou roztroušena do mnoha di-
639 menzí a tím odpadá možnost použití vzdálenostních funkcí. Zmíněné problémy shlukování
640 velkých dat řeší dvě techniky *metoda transformace rysů* a *metoda výběru atributů*. Pro

⁴Self–Organizing Map

⁵AGglomerative Nesting

⁶Clustering Using Wavelet Transformation

⁷Density–Based Spatial Clustering of Applications with Noise

641 efektivní shlukování je možné použít například algoritmus CLIQUE⁸.

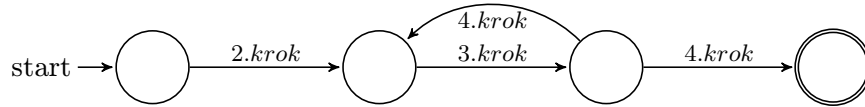
642 Metody rozkladu

643 Metody rozkladu rozdělují datové prvky do několika podmnožin, nazývané shluky. Počet
644 shluků musí být znám před zahájením samotného procesu. Přiřazení do konkrétních tříd
645 je, podle [28], jednoznačné nebo probíhá na základě míry příslušnosti objektů do shluků.
646 Pro velký počet objektů, se kterými se pracuje, jsou využívány různé iterační optimalizace.

647 Hlavním zástupcem u uvedených metod je algoritmus k -means, který je popsán níže.
648 Tvoří základ pro většinu metod shlukování nejen pro metody rozkladu. K dalším metodám
649 se řadí k -medoidů, k -modů, k -histogramů, fuzzy shluková analýza a další.

650 k -means

651 Shlukování pomocí algoritmu k -means je používáno pro data obsahující kvantitativní pro-
652 měnné a pro data, která nejsou příliš zašumělá. Základní proces je tvořen iterativním roz-
653 dělováním objektů do tříd na základě vzdáleností od jejich středů. Střed neboli centroid
654 shluku je vektor, jehož vzdálenost od součtu vzdáleností objektů v této třídě je minimální.
655 Celý tento proces je prezentován na obrázku 3.3 pomocí jednoduchého schématu konečného
656 automatu. Jednotlivé kroky konečného automatu odpovídají krokům k -means zobrazeného
pomocí algoritmu 3.1. Pro výpočet vzdáleností mezi objekty samotnými nebo mezi objekty



Obrázek 3.3: Algoritmus k -means zobrazený pomocí konečného automatu.

657 a středem je použita euklidovská vzdálenost⁹, která je vyobrazena na obrázku 3.2.

659 K hlavním výhodám algoritmu k -means patří jeho relativní efektivnost. Složitost algo-
660 ritmu je $O(TKN)$, kde N je počet objektů, K je počet shluků a T je počet iterací. Obvykle
661 platí, že počet objektů je mnohem větší než počet iterací i shluků. Na druhou stranu má i
662 řadu nevýhod, kvůli kterým je často různými způsoby modifikován (k -medoids, k -medians).
663 K hlavním „nedostatkům“ patří předem nutná znalost počtu shluků (tříd) K , do kterých
664 budou objekty zařazeny. Druhý často se vyskytující problém je samotné ukončení algo-
665 ritmu, které nastane u nalezení lokálního optima namísto optima globálního. Tato nepřes-
666 nost vzniká nevhodně zvoleným rozmístěním počátečních středů. Původní nemodifikovaná
667 verze algoritmu nedefinuje, jak se má postupovat, jsou-li nalezeny prázdné shluky.

668 K -menas je algoritmus, kterým jsou přiřazovány objekty (vektory) x_n , kde $n = 1, \dots, N$,
669 do S_k , kde $k = 1, \dots, K$, shluků. V prvním kroku jsou určeny počáteční středy tříd, do
670 kterých se budou objekty shlukovat. Určení počátečních centroidů c_k probíhá například
671 náhodným výběrem K objektů nebo K prvních objektů souboru. Druhým krokem jsou
672 zkoumány jednotlivé vzdálenosti objektů x_n od počátečních středů c_j pomocí euklidov-
673 ské vzdálenosti. Na základě nejmenší zjištěné vzdálenosti mezi objektem a centroidem je
674 objekt zařazen do shluku, kterému náleží právě tento střed. Ve třetím kroku, tak jako u
675 kroku prvního, jsou hledány nové středy shluků. Nyní již však nejsou zvoleny náhodně, ale

⁸CLustering In QUEst

⁹mean = střed, centroid je vektor průměrů

676 spočítány. Jsou vypočítány na základě průměrných jednotlivých hodnot objektů a uložen
677 jako m -rozměrný vektor. Čtvrtým krokem se algoritmus dostává do konečné fáze, kdy mo-
678 hou nastat dva možné případy. Nově nalezené středy nejsou příliš vzdáleny od předchozích
679 centroidů a proto je algoritmus ukončen. Druhá častěji se vyskytující možnost iterativně
680 provádí algoritmus od druhého kroku, dokud neplatí první možnost nebo dokud se objekty
681 nepřestanou přemísťovat úplně. Při popisu tohoto algoritmu bylo čerpáno z [28, 2]. Níže
682 zobrazený algoritmus 3.1 prezentuje krok po kroku metodu k -means.

-
1. náhodně zvol rozklad do K shluků
 2. urči centroidy pro všechny shluky v aktuálním rozkladu
 3. pro každý příklad x
 - 3.1 urči vzdálenosti $d(x, c_k)$, $k = 1, \dots, K$, kde c_k je centorid k -tého shluku
 - 3.2 nechť $d(x, c_l) = \min_k d(x, c_k)$
 - 3.3 není-li x součástí shluku l (k jehož centoridu c_l má nejblíže), přesuň x do shluku l
 4. došlo-li k nějakému přesunu, potom jdi na 2, jinak konec
-

Algoritmus 3.1: Metoda k -means byla převzata z [2].

683 Díky jednoduchosti a relativní rychlosti je metoda k -means stále výrazně využívána.
684 Uplatnění nachází v široké škále oblastí jako je například biologie nebo počítačová grafika.
685 Vzhledem k enormnímu počtu možného uspořádání nejsou výsledky vždy přesné, ale často
686 pouze přibližné.

687 3.4 Programy

688 V současné době na programovém trhu existuje mnoho systému, které jsou zaměřeny na data
689 mining. Mezi nejrozšířenější a nejdostupnější nástroje patří Weka a RapidMiner. K těmto
690 nástrojům je také možné zařadit program FIT-miner vyvíjený na fakultě informačních
691 technologií v Brně. V této práci byl pro samotný data miningg využit program RapidMiner,
692 který dostal přednost před ostatními. Z pohledu nástroje FIT-miner, který ve své základní
693 části podporuje z databází pouze Oracle, se RapidMiner jevil jako vhodnější. Kompatibilitu
694 pro databáze typu PostgreSQL již měl zabudovanou a tak nebylo potřeba vyvíjet žádné
695 doplňující moduly, jak by to bylo u FIT-mineru. V případě nástroje Weka, RapidMiner
696 působil propracovanějším dojmem a také nabízí lepší grafické zobrazení vyhodnocených
697 výsledků.

698 Dalším velmi rozšířeným a často používaným nástrojem je jazyk R . R vychází z jazyka
699 S , který ale není jako jazyk R volně šiřitelný. Statistický a grafický nástroj R je tedy volně
700 dostupným jazykem a prostředím, které je ovládáno pouze z příkazové řádky. Pro jednodušší
701 práci jej lze rozšířit o grafické rozhraní jako je RKWard nebo R Commander. Samotnou
702 aplikaci lze rozšířit o mnoho statistických doplňků, které jsou taktéž zdarma.

703 Mnoho programů pro dolování dat je založena na přístupu vizuálního programování.
704 Jedná se o proces, při kterém je uživatelem, za pomoci grafických prostředků, navržen
705 algoritmus a další postup práce. Jako příklad lze uvést poloprofesionální aplikaci *Orange*,
706 u které jsou nejdůležitější části psány pomocí C++ a rozšíření lze implementovat v jazyce
707 Python.

708 Na vybraných technicky zaměřených vysokých školách existují skupiny, které se zabývají
709 výzkumem a vývojem nástrojů pro data mining. Jako příklad lze uvést již dříve zmiňovaný
710 FIT-miner z fakulty informačních technologií v Brně nebo také projekt LISp-Miner z Vy-

711 soké školy ekonomické v Praze. LISp-Miner je otevřený akademický systém určený pro
712 výuku a výzkum metod pro dobývání znalostí z databází.

713 **RapidMiner**

714 RapidMiner je, tak jak je popsán na oficiálních stránkách produktu [27], celosvětově nej-
715 používanější open-source systém pro dolování dat. Je možné jej používat jako samotnou
716 aplikaci nebo jej začlenit jako komponentu do vlastních výrobků v podobě knihovny pro ja-
717 zyk Java. Pro zájemce je nabízen také ve verzích pro firmy, které jsou rozdílné v poplatcích,
718 podpoře pro zákazníka, záruce a dalších balíčků služeb zajišťující celkovou komplexnost a
719 spolehlivost produktu.

720 Jak již většina podobných aplikací, je v současné době implementován v jazyce Java,
721 díky které nabízí flexibilní nejen grafické prostředí. K vybraným základním rysům toho
722 nástroje, tak jak jsou prezentovány firmou *Rapid-i*, patří: výkonné, přesto intuitivní grafické
723 uživatelské rozhraní pro návrh procesů, jednoduché řešení pro transformaci dat, kontrola
724 výsledků již při samotném návrhu a další. Nástroj RapidMiner podporuje širokou škálu
725 metod a algoritmů pro data minig. Mnoho algoritmů je implementováno přímo v aplikaci,
726 ale také je použito metod z konkurenčního softwaru Weka. V základní verzi určené pro
727 veřejnost je k nalezení přes 100 procesů k modelování. Jsou zde zastoupeny jak metody
728 klasifikační a asociační, tak i metody shlukovací, z nichž lze jmenovat například DBSCAN,
729 k -medoids a hlavně k -means.

730 K dalším schopnostem RapidMineru je možnost spuštění jeho samotného pomocí gra-
731 fického rozhraní nebo z příkazové řádky. Jak již bylo uvedeno dříve, je také možné jej
732 použít jako knihovnu v jazyce Java. V této práci jsou použity první dvě možnosti. Pomocí
733 grafického prostředí byl vytvořen experiment, otestována jeho funkčnost a následně pro
734 jednotlivá shlukování použita šablona procesu, která byla volána z příkazové řádky. Tato
735 možnost je k dispozici díky tomu, že jsou projekty v programu RapidMiner ukládány do
736 čitelné a strukturované formy za pomoci značkovacího jazyka xml.

737 Kapitola 4

738 Implementace

739 Obsahem čtvrté kapitoly je popis praktické části této práce. Jsou zde charakterizovány
740 jednotlivé prvky, které byly použity jak pro získání dat, tak pro jejich následné uložení. V
741 první části je prezentována struktura architektury této práce. Její grafické znázornění je
742 ukázáno na obrázku 4.1.

743 Cílem této práce je dolování dat z Jabberu. Jak již bylo dříve napsáno, Jabber je real-
744 time síťová služba díky níž mohou její uživatelé komunikovat, informovat nebo sdílet svůj
745 status s jinými uživateli. Celá tato vzájemná komunikace skrývá rozsáhlé množství informací
746 o klientech dané sítě. Všechna tato navzájem vyměněná nebo poskytnutá data následně
747 poslouží jako zdroj samotnému dolování. Pro jejich uskladnění je využita databáze, jejichž
748 strukturální návrh prezentuje obrázek 4.2.

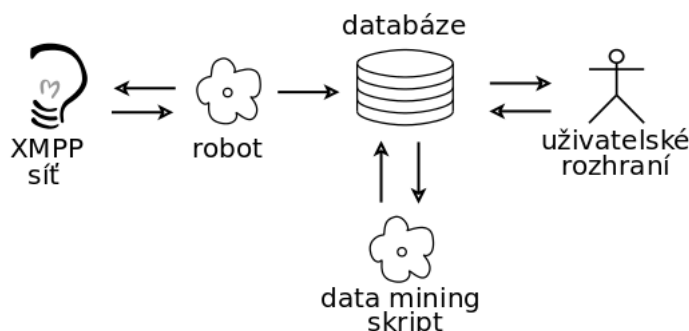
749 Třetí část této kapitoly je zaměřena na Jabber klienta neboli robota, který je imple-
750 mentován v jazyce C++ pouze s konzolovým rozhraním. Robot v této práci hraje roli
751 pasivního uživatele, který informace pouze přijímá. Ve vybraných případech dokáže uží-
752 vatele ze svého seznamu kontaktů vyzvat k zaslání odpovědi s informacemi na odpověď,
753 o kterou žádal. Struktura samotného robota je popsána níže a reprezentována obrázkem
754 4.3. V části o Jabber robotovi jsou také uvedeny informace o knihovně *gloox*, kterou je
755 zprostředkovány všechny náležitosti Jabber komunikace.

756 4.1 Architektura

757 Již ze samotného zadání a názvu této práce je zřejmé, že je třeba celý proces rozčlenit na
758 menší elementy. Vhodnou dekompozicí vzniklo pět jednotlivých ucelených prvků, které jsou
759 prezentovány schématem na obrázku 4.1. Konkrétně to jsou, zleva: *XMPP server*, *robot*, *da-*
760 *tabáze*, *data mining* a *uživatelské rozhraní*. Vzájemná výměna informací mezi jednotlivými
761 částmi je zobrazena pomocí šipek, které určují směr komunikace.

762 První část schématu, která je nazvaná XMPP síť, prezentuje obrázek 4.1. Tento oddíl je
763 obecně popsána v druhé kapitole. Vztah a vzájemná komunikace s robotem probíhá pomocí
764 internetové sítě, kdy XMPP síť souhrnně reprezentuje jednotlivé prvky, jako jsou Jabber
765 servery, uživatelé a jejich klienty. Jak je patrné, výměna informací mezi těmito částmi
766 probíhá obousměrně. Druhý element schématu *robot*, je charakterizován v podkapitole níže,
767 kde je podrobně popsán návrh jeho struktury. Jsou zde zdůrazněny jeho základní vlastnosti
768 a schopnosti. Dalším blokem je *databáze*, která reprezentuje datové úložiště. Do něhož jsou
769 za pomoci jednosměrného kanálu ukládána data, která jsou získávána z toku informací
770 proudícími mezi robotem a XMPP sítí. Přehled vybraných jednotlivých tabulek a jejich

771 atributů je možné nalézt v následující kapitole, která se zabývá návrhem databáze.
 772 Dalším oddílem, prezentovaným na obrázku 4.1 pod názvem *data mining skript*, je
 773 skript, který provádí samotný data minig. Jako první jsou vybraná data z databáze trans-
 774 formována do vhodného formátu pro dolování z dat. Tato část je podrobně popsána ve
 775 třetí kapitole, v oddílu zabývajícím se transformací dat. Přeformátovaná data jsou předána
 776 programu RapidMineru, kterým za pomoci algoritmu k -means je prováděn data mining.



Obrázek 4.1: Struktura architektury bakalářské práce.

777 Poslední částí je *uživatelské rozhraní*, které je implementováno pomocí webových služeb.
 778 Jako příklad lze uvést službu, která zobrazuje status uživatele na webové stránce. Uživateli
 779 pouze stačí vlastnit Jabber účet a mít přidáného tohoto robota do seznamu kontaktů. Od
 780 každého uživatele, jež má tohoto robota přidáného do svého seznamu kontaktů, je sbírána
 781 veškerá síťová aktivita. Možnosti, které mu nabízí uživatelské rozhraní v podobě webové
 782 prezentace tudíž záleží pouze na datech, které sám do sítě rozesílá. Jsou-li podporována
 783 všechna zde implementována rozšíření, která jsou uvedena v kapitole druhé, je možné využít
 784 jejich služeb prostřednictvím internetových stránek. Jako další příklad lze uvést zobrazení
 785 aktuálního umístění uživatele na mapách od firmy google, dle informací poskytnutých pro-
 786 střednictvím dokumentu User Geoloc [6].

787 4.2 Databáze

788 Data, která jsou sbírána robotem, jsou ukládána do objektově-relační databáze Postgre-
 789 SQL¹. PostgreSQL neboli také Postgres byl použit ve verzi 8.4.7 a je provozován na opera-
 790 čním systému Ubuntu.

791 Návrh databáze

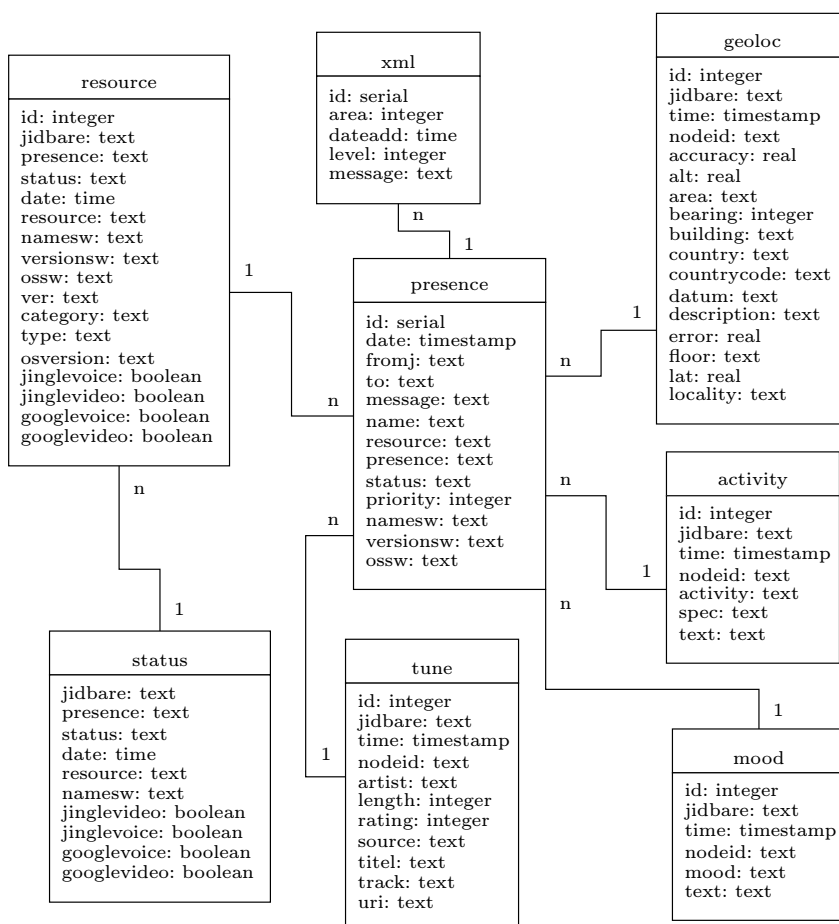
792 Struktura databáze, do které je ukládána veškerá komunikace Jabber robota, využívá rela-
 793 ční model. Obrázkem 4.2 jsou prezentovány nejdůležitější části databáze. Celou strukturu
 794 návrhu je možné nalézt v příloze F. V jednotlivých částech návrhu databáze je počítáno s
 795 druhotným využitím obsahu, které bude popsáno v dalších oddílech této práce.

796 V době návrhu databáze nebylo zcela zřejmé, která data budou následně analyzována.
 797 Z tohoto důvodu se struktura databáze snaží zachytit všechna „důležitá“ data. Za tímto
 798 účelem je v návrhu databáze obsažena tabulka *xml*, která je nositelem obsahu jak všech

¹vyvinul se z projektu Ingres

799 přijatých, tak i odeslaných zpráv. Tabulky *debug*, *level* a *logarea*, které v zúženém návrhu
800 databáze, uvedeném na obrázku 4.2, nejsou zobrazeny, jsou určeny pouze jako doplňkové
801 informace k typu zprávy. Tabulku *xml* je možné nahradit jednotlivými dalšími tabulkami,
802 které jsou zaměřeny na konkrétní data. Příkladem entity, která již obsahuje konkrétní data
803 bez xml prvků, je tabulka *message*, která je taktéž vyobrazena v příloze E. Jejím obsahem
804 jsou zprávy vzniklé při Jabber komunikaci mezi uživatelem a robotem, jehož schopnosti
805 budou popsány níže.

806 Tabulka *presence* z pohledu XMPP standardu prezentuje jeden ze tří základních částí
807 stanzy, element *presence*. Základním cílem této tabulky je shromažďování uživatelských
808 statusů. Jak již bylo zmíněno ve třetí kapitole, v části která se zabývá transformací, atribut
809 presence obsahuje hodnoty typu text. Příklad všech možných hodnot, kterých tento atri-
810 but může nabývat je prezentován v příloze E v části zabývající se elementem presence. K
811 dalším atributům této tabulky patří *message*, který je reprezentován textovým řetězcem a
812 rozšiřuje informace o stavu uživatele. V této části je často obsažen text, který je do statusu
813 přidán automaticky IM klientem. Transformovaný obsah této entity tvoří důležitou část při
814 získávání znalostí a to v podobě dat, ze kterých budou „dolovány“ informace.



Obrázek 4.2: Vybraná část struktury databáze.

815 Většina rozšíření standardu XMPP, která jsou popsána ve druhé kapitole, jsou pre-
816 zentována pomocí pěti tabulek. Konkrétně to jsou tabulky *geoloc*, *tune*, *mood*, *activity* a

817 *vcard*. Pro obsah a názvy atributů všech pěti tabulek s rozšířeními se staly vzory dokumenty
818 jednotlivých XEP, které je definují. Tabulky kromě těchto atributů obsahují také položku
819 *jidbare*, která, jak již název naznačuje, obsahuje pouze čisté JabberID bez resources. Tato
820 skutečnost vyplývá již ze samotného návrhu XEP protokolů. V tabulce *vcard* jsou také
821 položky, jejichž obsah je tvořen fotografiemi. Obrázky jsou zde uloženy ve stavu, tak jak
822 jsou přenášeny po síti, tedy pomocí datového formátu Base64. Base64 je algoritmus, který
823 převádí binární data na řetězec, který je tvořen pouze znaky ASCII tabulky.

824 S posledním rozšířením, nazvaném *Software version* je spjata tabulka resource, jejíž
825 několik sloupců tvoří informace o IM klientovi a operačním systému, na kterém běží. Tyto
826 základní údaje jsou definovány v protokolu [17], kde je také možné čerpat bližší informace.
827 Pokročilejší atributy, jako je *type* a *osversion* nacházejí svůj podklad v XEP dokumentu [7],
828 taktéž jako atribut *ver*. V neposlední řadě se zde nachází zmínka o podpoře audia a videa,
829 ať už v podobě *jingle* nebo *google*. Přesto tyto výše uvedené údaje nejsou hlavním důvodem
830 existence této tabulky. Její primární cíl je uchovávat informace o připojených uživateli a
831 všech jejich resources.

832 Entita resources tvoří základ pro tabulku *status*. Její obsah je automaticky generován z
833 obsahu tabulky popsané výše. Počet řádků odpovídá počtu uživatelů, kteří jsou v seznamu
834 kontaktů tohoto robota. Primárním cílem je informování o aktuálním stavu uživatele. Je
835 tedy poskytován stav, ve kterém se uživatel nachází s přihlédnutím na prioritu a resources.
836 Řádek entity obsahuje status a dostupnost uživatele a jeho resources s největší přihlášenou
837 prioritou.

838 4.3 Robot

839 V této části bude popsána aplikace, která zajišťuje real-time komunikaci s ostatními en-
840 titami Jabber sítě. Konzolový robot, který je vyvíjen pro operační systém Linux, také
841 vhodným způsobem pracuje s databází, kam jsou ukládána jednotlivá získaná data. Jabber
842 jádro síťové komunikace [23], je implementováno pomocí volně dostupné knihovny *gloox*. V
843 porovnání s jinými knihovnami disponuje lepší podporou a dokumentací. Gloox plně im-
844 plementuje standart XMPP Core [23] a z větší části i standard XMPP IM [24]. Dodatečně
845 je podporováno kolem třiceti XEP standardů mezi něž například patří *vcard-temp* [16] a
846 další.

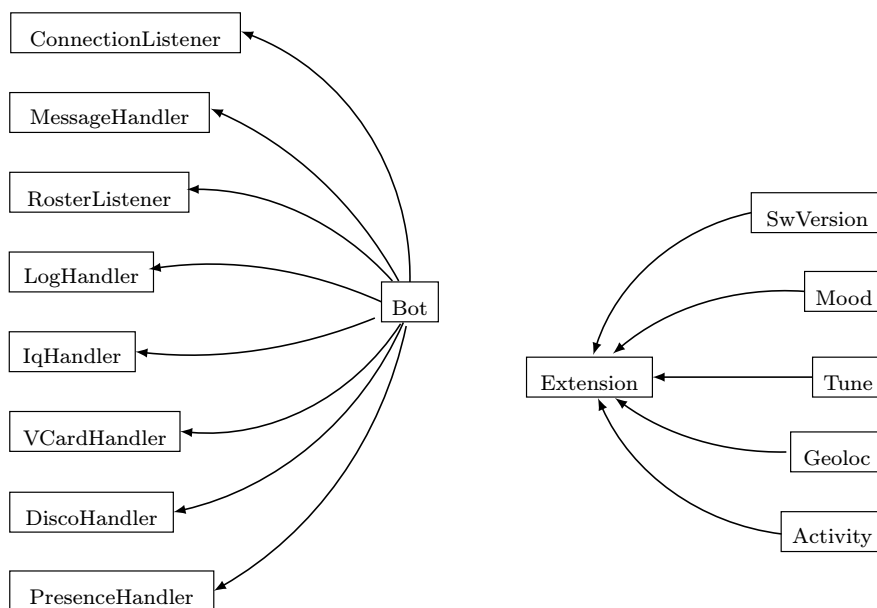
847 Při implementaci robota byly využity základní prvky objektově-orientovaného progra-
848 mování. Každá třída zde zastupuje jednoznačně oddělitelné elementy robota. Ať už se jedná
849 o blok komunikující s databází, různá rozšíření nebo o jádro robota. Všechny zde jmenované
850 objekty hrají důležitou roli jak v části dekompozice na menší ucelené celky tak i v celkové
851 komplexnosti programu. Jako příklad lze uvést třídy, které reprezentují jednotlivá pro tuto
852 práci vhodná rozšíření, jejichž základ je v podobě XEP dokumentů.

853 Návrh robota

854 Struktura návrhu tříd robota, která je v základní části prezentována obrázkem 4.3, je velmi
855 podobná struktuře návrhu databáze, obrázek 4.2. Mnoho atributů z robota a z databáze jsou
856 ve vztahu jedna ku jedné. Samotná komunikace a řízení mezi těmito entitami je zprostředko-
857 váno pomocí knihovny *libpq*. Libpq je programátorské rozhraní pro databázi PostgreSQL. Je
858 to sada funkcí v jazyce C, které umožňují klientským programům zadávat dotazy na server
859 a následně na ně obdržet odpovědi. Pomocí této knihovny je řízen přenos dat mezi robotem,
860 který sbírá data, a databází, kam jsou ukládána. Navázání spojení a následné jednotlivé

861 dotazy probíhají v blokujícím režimu. V souboru *connect.h* jsou, v podobě předdefinova-
 862 ných konstant, uvedeny všechny údaje nutné k navázání spojení ať už s databází nebo s
 863 komunikačním Jabber serverem. Tudiž se stává hlavní částí určenou k editaci připojení a
 864 Jabber účtu na němž robot běží. K dalšímu pouze textovému dokumentu patří *const.h*,
 865 který je zaměřen k definování jednotlivých příkazů a dotazů pro databázi. Jsou to příkazy
 866 pro tvorbu a aktualizaci tabulek, které jsou taktéž definované jako konstanty. Identifikační
 867 názvy, pro snadné odlišení, jsou psány velkými písmeny a začínají textem *DB_*.

868 Komunikace s databází se nachází v samostatné třídě, která zajišťuje samotnou prová-
 869 zanost mezi ní a daty. Pomocí funkcí jsou data, získaná z Jabber komunikace, přenášena a
 870 ukládána do jednotlivých tabulek databáze. Při zahájení činnosti robota je zajištěno navá-
 871 zání připojení k databázi a provedení nutných inicializačních kroků. Jako je tvorba nových
 872 tabulek, při prvním zapnutí aplikace, nebo kontrola existence těchto entit.



Obrázek 4.3: Vybraná část struktury robota.

873 Pro rozšíření podle dokumentů XEP popsanych v kapitole druhé, v části zabývající se
 874 implementovanými rozšířeními, je základ nadtřída *Extension*. Tato třída obsahuje základní
 875 funkce a parametry pro všechna rozšíření. Jako příklad lze uvést uživatelské jméno klienta,
 876 který dané rozšíření „šíří“ po síti. Třídy, které z *Extension* dědí a jsou prezentovány na
 877 návrhu robota uvedeném na obrázku 4.3, jsou následující: *Geoloc*, *Tune*, *Mood*, *Activity* a
 878 *SwVersion*. Samotnou strukturou těchto rozšíření, jak již bylo uvedeno výše, jsou pomocí
 879 parametrů kopírovány vlastnosti jednotlivých XEP dokumentů. Tedy ke každému atributu
 880 xml zprávy existuje jak proměnná uchovávající její obsah tak i tzv. „get“ a „set“ metoda.
 881 Pomocí další metody jsou jednotlivé části rozšíření rozparsovány, uloženy do tříd a následně
 882 vloženy do databázi. Poslední doposud nezmiňované rozšíření v podobě *vcard*, je využito z
 883 knihovny *gloox*, která jej implementuje.

884 Základní část, kterou je možné považovat za jádro aplikace, je třída *Bot*. Tato třída je
 885 založena na již zmiňované C++ knihovně *gloox*. Je implementována za pomoci vybraných
 886 tříd, ze kterých dědí. Jsou to především „handler“ třídy, kterými je zajišťován přístup k
 887 jednotlivým zprávám. Nebo-li jejich prostřednictvím je získán přístup k samotným elemen-

888 tím stanzy, jako je message, iq, a presence, jejichž vlastnosti a využití jsou popsány v druhé
889 kapitole. Konkrétní seznam tříd, ze kterých je děděno, zobrazuje návrh robota na obrázku
890 4.3. Jako příklad lze uvést třídu, z prostoru jmen gloox, *VcardHandler*, díky níž je možné
891 získat a zpracovávat vcard uživatelů, kteří jsou v seznamu kontaktů robota. Samotné přidá-
892 vání uživatelů do seznamu kontaktů je prováděno automaticky. Pouze na straně klienta je
893 nutné požádat robota o autorizaci, která však proběhne automaticky. Jednotlivé kontakty
894 nejsou žádným způsobem tříděny do skupin, tudíž existuje pouze jedna.

895 Příkazy

896 K dalším schopnostem robota, tedy kromě sbírání dat a jejich následného ukládání do da-
897 tabáze, patří reagovat na vybrané zprávy. Tyto zprávy jsou přijímány pouze od uživatelů,
898 kteří byli začleněni do seznamu kontaktů tohoto robota. Základní příkaz, který nesmí chy-
899 bět u žádné aplikace, je získání nápovědy. V první řadě informuje o verzi robota a jeho
900 základních vlastnostech. Dále je odeslán stručný ale výstižný seznam příkazů i s jejich
901 vysvětlením.

Přehled základních zpráv a jejich vysvětlení je prezentováno v tabulce 4.1. Kde první

příkaz	vysvětlení
HELP	Přehled příkazů a jejich popis.
INFO	Popis aplikace.
HISTORY	Výpis posledních 10 změn statusů.
TUNE	Výpis o přehrávané hudbě.
GEOLOC	Výpis o geografické poloze.
MOOD	Výpis o aktuální náladě.
ACTIVITY	Výpis o aktuální činnosti.
VCARD	Základní obsah VCard

Tabulka 4.1: Přehled příkazů pro robota.

902
903 sloupec značí příkaz, který rozlišuje velká a malá písmena, a druhý jej krátce a výstižně
904 popisuje. V případě, že uživatel nemá žádný údaj v databázi potřebný k vyhodnocení
905 dotazu, je zaslána pouze informativní zpráva, které vzniklý problém vysvětluje. V této
906 části robota je skryt velký potenciál pro další vývoj. Více informací je uvedeno v následující
907 kapitole, která je celá věnována možným rozšířením a pokračováním této práce.

908 Kapitola 5

909 Pokračování práce

910 V současné době, která je velmi uspěchaná a plná moderních technologií, je velmi důležité
911 být informován. S tímto také souvisí snaha poskytovat informace svému okolí. Díky pro-
912 tokolu XMPP a real-time komunikačních sítí je uspokojení potřeb, jako je získávání in-
913 formací a jejich zpětné poskytování v oblasti elektronické komunikace, snadno řešitelné.
914 Existuje rozsáhlé množství různých sociálních sítí a funkcí, které jsou uživatelům nabízeny.
915 Poskytované služby sociálními sítěmi jsou využívány ve značném množství. Služby, jako
916 jsou prezentování vlastního statusu v klientovi prostřednictvím elementu presence, jsou již
917 považovány za samozřejmost. Tato funkčnost je nabízena u všech tzv. „instant messaging“
918 služeb. Jabber jde v této oblasti ještě dál a díky protokolům *User Mood* [21] a *User Activity*
919 [11] je možné sdělit nejen svou náladu ale také informace o aktuálně prováděné činnosti.

920 Většina služeb, které jsou dostupné na internetu a jsou podporované různými servery,
921 poskytuje pouze informace o aktuálním stavu klienta. Pomocí výše uvedených standardů
922 by bylo možné připojit k základní webové prezentaci statusu i informace rozšířené. Také za-
923 pojení standardu *Entity Capabilities* [7], který je popsán na konci druhé kapitoly, by mohlo
924 přinést nové pokročilejší, užitečné informace. Konkrétně je to samotná podpora IM apli-
925 kací v oblasti dalších XEP dokumentů. Nezasvěcený uživatel by tímto velice jednoduchým
926 rozšířením dokázal zjistit seznam služeb podporovaných jeho IM programem. Ať už by se
927 jednalo o seznam funkcí, které jsou pouze přijímány, nebo výčet rozšíření, k jejich odběru
928 je zaregistrován. Jako příklad lze uvést zjištění schopnosti, ať už přijímání nebo odesílání
929 informací o přehrávané hudbě, popsané dokumentem *User Tune* [18].

930 Další oblast, ve které lze nalézt možnost rozšíření, se také zabývá webovou prezentací.
931 Nyní však jde již konkrétně o robota vytvořeného k účelu této práce. Informace potřebné k
932 zobrazení aktuálního stavu klienta na internetové stránce, jsou uloženy v databázi v tabulce
933 status. Tato tabulka je automaticky generována z obsahu jiné entity, která obsahuje údaje
934 o všech připojeních daného uživatele. Vše pracuje správně až do té chvíle, kdy uživatel
935 svou IM aplikaci neukončí korektně. Nastává situace, kdy na server není zaslána zpráva,
936 která informuje o změně statusu. Díky tomuto neočekávanému ukončení klienta je uživatel
937 prezentován, jako by byl připojen i když je tomu naopak. Zdrojová tabulka se v této chvíli
938 stává neaktuální. Navrhované řešení této situace využívá rozšíření *XMPP ping* [20]. Díky
939 němuž by se mělo dát zjistit, zda je uživatel stále dostupný nebo již nikoliv.

940 V oddílu robota by bylo možné pokračovat v části, která se zabývá příkazy zaslané ze
941 strany uživatele. S tímto rozšíření úzce souvisí rozčlenění uživatelů do jednotlivých skupin.
942 Každá takováto část seznamu kontaktů by vlastnila různá práva, která by se stala zákla-
943 dem pro poskytování konkrétních služeb. Zařazení do těchto skupin by mohlo probíhat při
944 žádosti o počáteční autorizaci, která by obsahovala zprávu s přesně definovaným řetězcem.

945 Funkce, která je již uvedena výše a to v podobě rozšíření webové prezentace, by také mohla
946 být poskytována prostřednictvím robota. Konkrétně se jedná o zjištění podporovaných ať
947 již přijímaných nebo odesílaných rozšíření. Uživatel by pouze poslal robotovi striktně před-
948 definovanou zprávu a jako odpověď by obdržel seznam podporovaných funkcí.

949 V měsíci březen, tohoto roku, vyšla nová verze základních dokumentů RFC. V této
950 práci, již ale z těchto standardů nejsou zahrnuty žádné změny. V době vydání se totiž práce
951 dostával do konečné fáze vývoje. Proto změny přinesené novými protokoly by taktéž mohly
952 být uplatněny při případném pokračování této práce.

953 Kapitola 6

954 Vyhodnocení výsledků

955 6.1 Manuální rozbor dat

956 6.2 ...

957 Kapitola 7

958 Závěr

959

Literatura

- [1] Adams, D.: *Programming jabber*. Sebastopol: O'Reilly, první vydání, 2002, 455 s., ISBN 05-960-0202-5.
- [2] Berka, P.: *Dobývání znalostí z databází*. Praha: Academia, první vydání, 2003, 366 s., ISBN 80-200-1062-9.
- [3] Bramer, M.: *Principles of Data mining*. London: Springer, první vydání, 2007, 343 s., ISBN 18-462-8765-0.
- [4] Fayyad, U. M.; Smyth, P.: *Advances in knowledge discovery and data mining*. California: MIT Press, první vydání, 1996, 611 s., ISBN 02-625-6097-6.
- [5] Han, J.; Kamber, M.: *Data mining : concepts and techniques*. San Francisco: Morgan Kaufmann Publisher, druhé vydání, 2006, 770 s., ISBN 15-586-0901-6.
- [6] Hildebrand, J.; Saint-Andre, P.: XEP-0080: User Location. [online], 15-09-2009, [cit. 9. května 2011].
URL <http://xmpp.org/extensions/xep-0080.html>
- [7] Hildebrand, J.; Saint-Andre, P.; Tronçon, R.; aj.: XEP-0115: Entity Capabilities. [online], 26-02-2008, [cit. 9. května 2011].
URL <http://xmpp.org/extensions/xep-0115.html>
- [8] Hruška, T.: *Informační systémy : IIS/PIS*. Brno: Fakulta informačních technologií, 2008, 14733 s.
- [9] Kolektiv autorů: Extensible Markup Language (XML) 1.0. [online], 26-11-2008, [cit. 9. května 2011].
URL <http://www.w3.org/TR/2008/REC-xml-20081126/>
- [10] Kosek, J.: *XML pro každého : podrobný průvodce*. Praha: Grada, první vydání, 2000, 163 s., ISBN 80-716-9860-1.
- [11] Meijer, R.; Saint-Andre, P.: XEP-0108: User Activity. [online], 29-10-2008, [cit. 9. května 2011].
URL <http://xmpp.org/extensions/xep-0108.html>
- [12] Millard, P.; Saint-Andre, P.; Meijer, R.: XEP-0060: Publish-Subscribe. [online], 12-07-2010, [cit. 9. května 2011].
URL <http://xmpp.org/extensions/xep-0060.html>

- 990 [13] Mizzi, S.; Saint-Andre, P.: XEP-0292: vCard4 Over XMPP. [online], 02-26-2008, [cit.
991 9. května 2011].
992 URL <http://xmpp.org/extensions/xep-0292.html>
- 993 [14] Moore, D.; Wright, W.: *Jabber developer's handbook*. Indianapolis: Sams Publishing,
994 první vydání, 2004, 487 s., iISBN 06-723-2536-5.
- 995 [15] Nemrava, M.; Pospíšil, J.: Dolování dat a jeho aplikace. [online], 2006, [cit. 9. května
996 2011].
997 URL http://www.spatial.cs.umn.edu/paper_ps/dmchap.pdf
- 998 [16] Saint-Andre, P.: XEP-0054: vcard-temp. [online], 07-16-2008, [cit. 9. května 2011].
999 URL <http://xmpp.org/extensions/xep-0054.html>
- 1000 [17] Saint-Andre, P.: XEP-0092: Software Version. [online], 02-15-2007, [cit. 9. května
1001 2011].
1002 URL <http://xmpp.org/extensions/xep-0092.html>
- 1003 [18] Saint-Andre, P.: XEP-0118: User Tune. [online], 30-01-2008, [cit. 9. května 2011].
1004 URL <http://xmpp.org/extensions/xep-0118.html>
- 1005 [19] Saint-Andre, P.: XEP-0153: vCard-Based Avatars. [online], 16-08-2006, [cit. 9. května
1006 2011].
1007 URL <http://xmpp.org/extensions/xep-0153.html>
- 1008 [20] Saint-Andre, P.: XEP-0199: XMPP Ping. [online], 03-06-2009, [cit. 9. května 2011].
1009 URL <http://xmpp.org/extensions/xep-0199.html>
- 1010 [21] Saint-Andre, P.; Meijer, R.: XEP-0107: User Mood. [online], 29-10-2008, [cit.
1011 9. května 2011].
1012 URL <http://xmpp.org/extensions/xep-0107.html>
- 1013 [22] Saint-Andre, P.; Smith, K.: XEP-0163: Personal Eventing Protocol. [online],
1014 12-07-2010, [cit. 9. května 2011].
1015 URL <http://xmpp.org/extensions/xep-0163.html>
- 1016 [23] Saint-André, P.: Extensible Messaging and Presence Protocol (XMPP): Core.
1017 [online], 10-2004, [cit. 9. května 2011].
1018 URL <http://tools.ietf.org/html/rfc3920>
- 1019 [24] Saint-André, P.: Extensible Messaging and Presence Protocol (XMPP): Instant
1020 Messaging and Presence. [online], 10-2004, [cit. 9. května 2011].
1021 URL <http://tools.ietf.org/html/rfc3921>
- 1022 [25] Saint-André, P.; Smith, K.; Troncon, R.: *XMPP : the definitive guide : building
1023 real-time applications with jabber technologies*. Sebastopol: O'Reilly, první vydání,
1024 2009, 287 s., iISBN 978-059-6521-264.
- 1025 [26] WWW Stránky: Database Systems. [online], 2007, [cit. 9. května 2011].
1026 URL
1027 http://www.cs.uct.ac.za/mit_notes_devel/Database/Lates%t/index.html

- 1028 [27] WWW Stránky: RapidMiner. [online], 2011, [cit. 9. května 2011].
1029 URL <http://rapid-i.com/content/view/181/190/>
- 1030 [28] Řezánková, H.; Húsek, D.; Snášel, V.: *Shluková analýza dat*. Praha: Professional
1031 Publishing, druhé vydání, 2009, 218 s., iISBN 978-808-6946-818.

¹⁰³² **Příloha A**

¹⁰³³ **Obsah CD**

¹⁰³⁴ **Příloha B**

¹⁰³⁵ **Manual**

¹⁰³⁶ **Příloha C**

¹⁰³⁷ **Konfigurační soubor**

1038 Příloha D

1039 Slovník zkratek

1040 **Base64** — převedení binárních dat pomocí znaků ASCII.

1041 **GPG** (GNU Privacy Guard) — slouží k podepisování a kontrolu podpisu různých dat.

1042 **ID3v1** — datový formát obsahující informace o skladbě.

1043 **IM služby** (Instant messaging) — umožňují posílat zprávy a chatovat mezi uživateli.

1044 **Jabber** — viz XMPP.

1045 **JEP** (Jabber Enhancement Proposal) — starší název pro XEP.

1046 **JID** (Jabber ID) — jednoznačný identifikátor v síti Jabber.

1047 **SASL** (Simple Authentication and Security Layer) — metoda sloužící ověřování klientů
1048 na serverech.

1049 **Stanza** — XML stream.

1050 **SQL** (Structured Query Language) — jazyk používaný pro komunikaci s databázemi.

1051 **TLS** (Transport Layer Security) — kryptografický protokol, poskytuje zabezpečenou ko-
1052 munikaci na internetu.

1053 **TSQL** (Transact-SQL) — jazyk pro temporální databáze.

1054 **vCard** — elektronická osobní vizitka.

1055 **XEP** (XMPP Extension Protocol) — rozšiřuje protokol RFC.

1056 **XML** (Extensible Markup Language) — univerzální značkovací jazyk.

1057 **XMPP** (Extensible Messaging and Presence Protocol) — protokol pro doručování zpráv.

1058 Příloha E

1059 Stanza - základní schéma

1060 Přehled základních elementů, které jsou využívány při Jabber komunikaci. Struktura jed-
1061 notlivých částí stanzy ukazuje pouze prvky relativní k této práci. Pomocí hranatých závorek
1062 je znázorněna množina, ze které musí být vybrán právě jeden prvek. Na místě uvozovek se
1063 očekává jakákoliv povolená hodnota.

1064 Iq

```
1      <iq from=""  
2          to=""  
3          type="[ get , set , result , error ]"  
4          id=""  
5          Namespace  
6      </iq>
```

Algoritmus E.1: Popis elementu *iq*.

1065 Message

```
1      <message from=""  
2          to=""  
3          type="[ normal , chat , groupchat , headline , error ]"  
4          id=""  
5          <body> </body>  
6          <x xmlns="jabber:x:event">  
7              [ Offline , Delivered , Displayed , Composing ]  
8          <subject> </subject>  
9          <thread> </thread>  
10         <error> </error>  
11         <x> </x>  
12     </message>
```

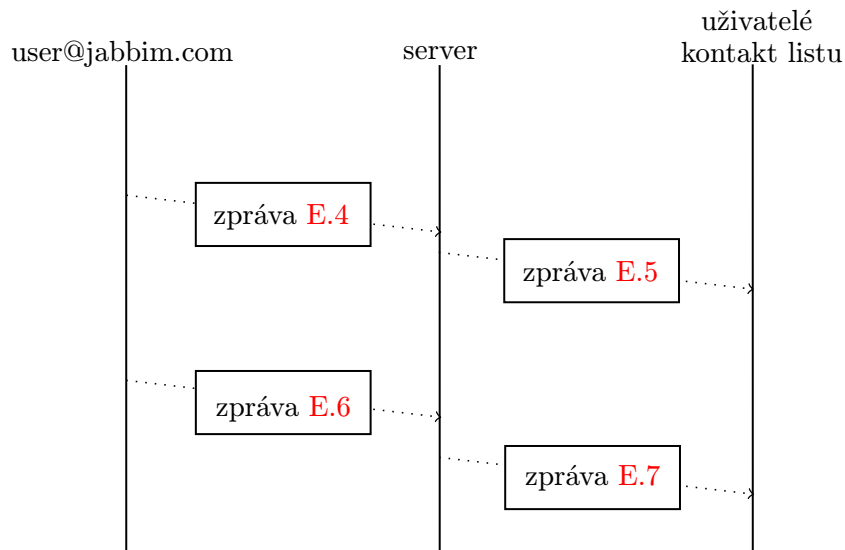
Algoritmus E.2: Popis elementu *message*.

```
1  <presence from=""
2      to=""
3      type="[available, unavailable, probe, subscribe,
4      unsubscribe, subscribed, unsubscribed, error]"
5      id=""
6  <show>
7      [away, chat, dnd, normal, xa]
8  </show>
9  <status>    </status>
10 <priority>   </priority>
11 <error>      </error>
12 </presence>
```

Algoritmus E.3: Popis elementu *presence*.

1067 Přehled průběhu rozšíření

1068 Ukázka celého příkladu šíření statusu pomocí rozšíření *User Tune*. Uživatel *user* poslouchá hudbu a informuje server zasláním zprávy zobrazené v příkladu E.4.



Obrázek E.1: Ukázka „šíření“ *User Tune*.

1069

```

1  <iq from="user@jabbim.com" type="set" id="pub1">
2    <pubsub xmlns="http://jabber.org/protocol/pubsub">
3      <publish node="http://jabber.org/protocol/tune">
4        <item>
5          <tune xmlns="http://jabber.org/protocol/tune">
6            <artist>Daniel Landa</artist>
7            <length>255</length>
8            <source>Nigredo</source>
9            <title>1968</title>
10           <track>5</track>
11          </tune>
12        </item>
13      </publish>
14    </pubsub>
15  </iq>

```

Algoritmus E.4: Informování serveru o právě přehrávané hudbě.

1070 Server obdrží informace od klienta *user* zprávu o přehrávací hudbě. Pomocí elementu
1071 *message* ji přepoše všem uživatelům z kontakt listu uživatele *user*, kteří jsou pro odběr
těchto typů zpráv zaregistrováni. Tato struktura zprávy je prezentována na příkladu E.5.

```
1  <message from="user@jabbim.com" type="set"
2      to="jabinfo@jabbim.com/bot" id="pub1">
3      <event xmlns="http://jabber.org/protocol/pubsub#event">
4          <items node="http://jabber.org/protocol/tune">
5              <item>
6                  <tune xmlns="http://jabber.org/protocol/tune">
7                      <artist>Daniel Landa</artist>
8                      <length>255</length>
9                      <source>Nigredo</source>
10                     <title>1968</title>
11                     <track>5</track>
12                 </tune>
13             </item>
14         </items>
15     </event>
16 </message>
```

Algoritmus E.5: Server informuje uživatele podporující rozšíření o stavu *user@jabbim.com*.

1072 Zpráva o přehrávané hudbě je také přeposlána všem otevřeným spojením uživatele *user*,
1073 ukázáno na příkladě E.6.
1074

```
1  <message from="user@jabbim.com" type="set"
2      to="user@jabbim.com/doma" id="pub2">
3      <event xmlns="http://jabber.org/protocol/pubsub#event">
4          <items node="http://jabber.org/protocol/tune">
5              <item>
6                  <tune xmlns="http://jabber.org/protocol/tune">
7                      <artist>Daniel Landa</artist>
8                      <length>255</length>
9                      <source>Nigredo</source>
10                     <title>1968</title>
11                     <track>5</track>
12                 </tune>
13             </item>
14         </items>
15     </event>
16 </message>
```

Algoritmus E.6: Server přepoše informace o přehrávané hudbě všem otevřeným spojením
uživatele *user@jabbim.com*.

1075 Přestane-li uživatel *user* poslouchat/vysílat informace o přehrávané hudbě, provede to
1076 pomocí zprávy ukázané na příkladu E.7. Zpráva typu *iq*, ve které je položka *tune* nesoucí
informace o skladbě prázdná.

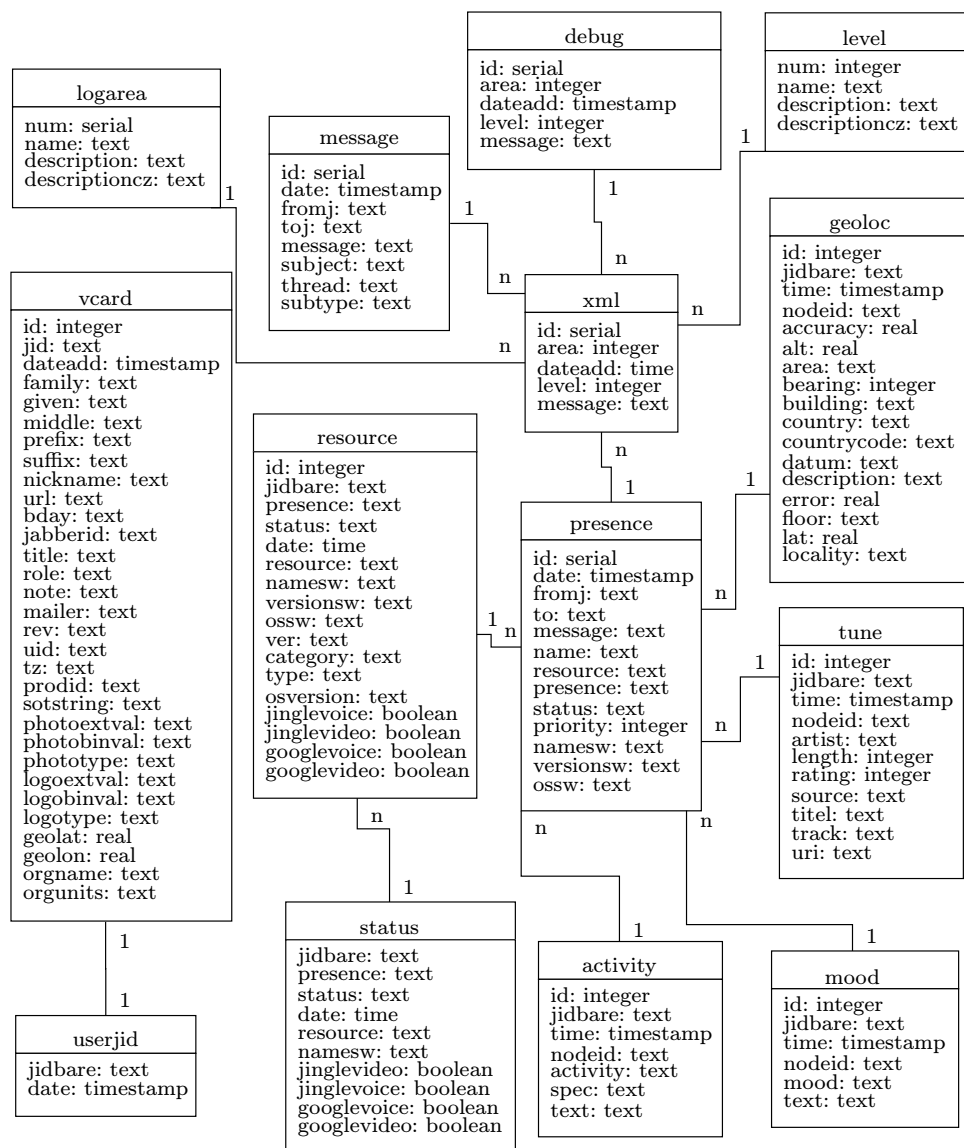
```
1 <iq from="user@jabbim.com/prace" type="set" id="pub1">
2   <pubsub xmlns="http://jabber.org/protocol/pubsub">
3     <publish node="http://jabber.org/protocol/tune">
4       <item>
5         <tune xmlns="http://jabber.org/protocol/tune"/>
6       </item>
7     </publish>
8   </pubsub>
9 </iq>
```

Algoritmus E.7: Uživatel ukončil „vysílání“ rozšířených zpráv o svém stavu.

1077
1078 Server informuje všechny účastníky odběru zprávou, která má položku *tune* prázdnou.
1079 Tak jak to prezentuje příklad E.8.

```
1 <message from="user@jabbim.com"
2   to="jabinfo@jabbim.com/bot">
3   <event xmlns="http://jabber.org/protocol/pubsub#event">
4     <items node="http://jabber.org/protocol/tune">
5       <item>
6         <tune xmlns="http://jabber.org/protocol/tune"/>
7       </item>
8     </items>
9   </event>
10 </message>
```

Algoritmus E.8: Server informuje klienty o ukončení šíření rozšířeného statusu uživatele *user@jabbim.com*.



Obrázek F.1: Struktura databáze