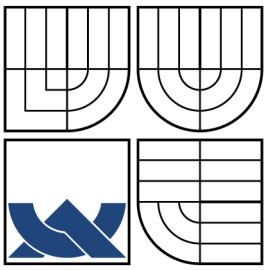
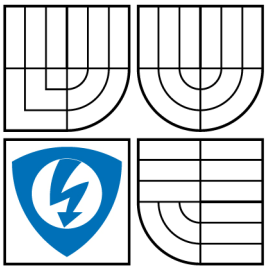


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

# KOMUNIKAČNÍ SOFTWARE PRO TERMINÁLOVÉ KLIENTY LINUXOVÉHO SERVERU

COMMUNICATION SOFTWARE FOR TERMINAL CLIENTS OF A LINUX SERVER

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

AUTOR PRÁCE — BC. KAREL HANÁK
AUTHOR

VEDOUCÍ PRÁCE — ING. MOJMÍR JELÍNEK
SUPERVISOR

BRNO 2009

**MÍSTO TOHOTO LISTU BUDE VLOŽENO**
**ZADÁNÍ DIPLOMOVÉ PRÁCE**

# LICENČNÍ SMLOUVA
# POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

**1. Pan/paní**

Jméno a příjmení: Bc. Karel Hanák
Bytem: náměstí Míru 103/20, 568 02 Svitavy
Narozen/a (datum a místo): 9.1.1980, Svitavy
(dále jen „autor“)

a

**2. Vysoké učení technické v Brně**

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 244/53, 602 00, Brno
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Ing. Kamil Vrba, CSc.
(dále jen „nabyvatel“)

## Čl. 1
## Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   - □ disertační práce
   - □ diplomová práce
   - □ bakalářská práce
   - □ jiná práce, jejíž druh je specifikován jako .....

   (dále jen VŠKP nebo dílo)

Název VŠKP: Komunikační software pro terminálové klienty linuxového serveru

Vedoucí/ školitel VŠKP: Ing. Mojmír Jelínek

Ústav: Ústav telekomunikací

Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:

- □ tištěné formě – počet exemplářů ……..
- □ elektronické formě – počet exemplářů ……..

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

## Článek 2
## Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti
   - □ ihned po uzavření této smlouvy
   - □ 1 rok po uzavření této smlouvy
   - □ 3 roky po uzavření této smlouvy
   - □ 5 let po uzavření této smlouvy
   - □ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

## Článek 3
## Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: …….

…….. ……

Nabyvatel Autor

# ANOTACE

V práci je obsažen návrh a realizace prostředí vhodného pro provoz síťových aplikací klientů, jenž používají běžné terminály. Jsou zde realizovány příklady s jejichž pomocí je prezentován způsob využití prostředí. Vychází se zde z centralizovaného způsobu komunikace. Toho je využito i jako možnosti navázání na řídící podsystémy, tedy neomezené možnosti regulace řízení systémů budov a přístup ke spotřebičům, práva uživatelů přistupovat přes datové body k zařízením. Prostředí je vystaveno na operačním systému Linux a databázi MySQL. Jeho nasazení se předpokládá na serveru, tedy v síťovém prostředí. S tím je spojena i celková bezpečnostní politika a zapracováno je i sociální ošetření možností klientů.

# KLÍČOVÁ SLOVA

úzký klient, virtuální prostředí, datový bod, řídící podsystém, Linux, terminál, skupiny uživatelů, MySQL databáze

# ABSTRACT

The thesis contains a proposal and implementation of an environment convenient for operation of network client applications which use common terminals. It also consists of implemented examples where the way of their usage is presented. The centralized way of communication is the basis. The approach is used also for the possibility of their joining with managing subsystems, i.e. unlimited ways of regulation of systems for real estate management, access to devices, user authority access to access data points to the devices. The environment is based on operation system Linux and database MySQL. Their realization is supposed on a server, in the network environment. This relates also to the overall security policy and this work also focused on social treatment of clients possibilities.



# KEYWORDS

thin client, virtual environment, data point, control subsystem, Linux, terminal, group of users, MySQL database

# BIBLIOGRAFICKÁ CITACE

HANÁK, K. Komunikační software pro terminálové klienty linuxového serveru. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 48 s. Vedoucí diplomové práce Ing. Mojmír Jelínek.

# PROHLÁŠENÍ

Prohlašuji, že svoji diplomovou práci na téma „Komunikační software pro terminálové klienty linuxového serveru“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne .....

.....

Podpis autora

# PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce Ing. Mojmíru Jelínkovi, doktorandovi Ústavu telekomunikací na FEKT VUT v Brně, za velmi užitečnou metodickou pomoc a cenné rady při zpracování diplomové práce.

V Brně dne .....

.....

Podpis autora

# Seznam použitých zkratek, veličin a symbolů

| | |
|---|---|
| AI | Datový bod analogového vstupu |
| AO | Datový bod analogového výstupu |
| app | Aplikace |
| Bash | Verze Bourne Again jazyka Shell |
| CD | Kompaktní disk |
| DI | Datový bod číslicového vstupu |
| DO | Datový bod číslicového výstupu |
| DP | Obecný datový bod |
| E-R | Diagram znázorňující tabulky databáze |
| GNU | Všeobecná veřejná licence |
| GUI | Grafické uživatelské prostředí |
| I | Veličina elektrického proudu |
| ID | Identifikační číslo |
| Ni1000, Ni10000 | Normalizovaná charakteristika čidla |
| PHP | Hypertextový preprocesor |
| PK | Primární klíč |
| Pt100, Pt1000 | Normalizovaná charakteristika čidla |
| s, s1, s2, s3, s4 | Objekt značící místo v Petriho síti |
| Shell | Název příkazového procesoru |
| SSH | Secure Shell – šifrovaná alternativa Telnet |
| SQL | Structured Query Language – strukturovaný dotazovací jazyk |
| U | Veličina elektrického napětí |
| UML | Unified Modeling Language – grafický model programu |
| t, t1, t2 , t3 | Objekt značící přechod v Petriho síti |
| .sh | Přípona souboru se skriptem jazyka Shell |
| .sql | Přípona souboru s dávkou příkazů jazyka SQL |

# Obsah

# Seznam obrázků

# 1 Úvod

Tato diplomová práce se zabývá problematikou a řeší situaci, kdy je vyžadováno na straně klienta, jenž se připojuje pomocí terminálu, využívat k maximální spokojenosti a s nejvyšší možnou efektivitou a všemi možnostmi programy na straně serveru. Dále předvést ukázky možných programů za účelem prezentace využití a funkčnosti tohoto řešení.

V celém řešení jsou použity pouze programy a technologie s otevřenou licenční politikou. Důvodem je především snaha o volné možnosti použití, nevázanosti na proprietální software a formáty, a především prozřetelnost v otázce dalšího možného bádání nijak vázanému na komerční vlivy nebo situace v oblasti software.

## 1.1 Struktura práce

Práce je dle kapitol členitě rozdělena do několika částí. Ze začátku se zabývá úrovní hardware, to znamená možnostmi zásahu programu do elektrických veličin, případně snímání hodnot z reálného světa. Předpokládá se využití programu například pro průmyslové využití nebo v oblasti ovládání elektroniky nebo řízení systémů v domácnosti.

Dále se naznačuje symbolický návrh programu na základě pravidel teoretické informatiky. Tato pravidla umožňují zdárnou realizaci na základě úspěšného návrhu programu z hlediska podchycení jeho výjimek a tím zaručují bezchybný provoz.

Samotná struktura programu udává logiku pro přístup k uživateli a poskytuje mu prostředí pro provoz vlastních aplikací. A zaopatřuje především technické zázemí. Dále zprostředkovává zázemí manipulace s finančním kreditem, tedy obdobu komerčních bank.

V hlavní části práce jsou detailně vysvětleny jednotlivé části programu na ukázkových příkladech použití, jako je řešení přístupových práv a přístup do databáze v různých situacích.

Na konci práce jsou uvedena srovnání s dalšími alternativními prostředky. Práce se zde zařazuje mezi podobné projekty a poukazuje na rozdíly a obhajuje vlastní řešení. V následující kapitole je poukázáno na možnosti dalšího rozšíření projektu. Tím jsou i vymezeny možnosti a oblasti, kterými může být práce dále směřována.

## 1.2 Praktické využití

Hlavní tendencí je soustředit veškeré dílčí úkony této diplomové práce obecně tak, aby bylo možné jejich všestranné využití v širokém spektru oblastí a s dalšími neuzavřenými možnostmi rozšíření. Každý krok této diplomové práce bude prezentován v ukázkových programech.

Tyto ukázkové aplikace neprezentují svoje vlastnosti jako nejlepší možná řešení, ale objasňují vlastnosti komunikačního programu, jenž jsou v těchto ukázkových příkladech programů využity a možností, k nimž se tyto principy dále využít dají.

## 1.3 Možnosti řešení

Princip užití unixového terminálu není nezbytným řešením. Staví se do role alternativy k dalším způsobům, jenž se dnes využívají. Častým způsobem a významným konkurentem může být například technologie na bázi webových aplikací s využitím dynamických možností dnešního internetu, například technologie skriptovacího jazyka PHP. [15]

Způsob terminálového připojení v tomto kontextu zdánlivě nepřináší žádný významný rozdíl nebo závažné vylepšení. Rozdíl se však naskytne, když zvážíme další možnosti rozšíření a udávaný směr obecného využití zařízení.

## 1.4 Principy úzkého připojení

Zařízení se dá z jednoho hlediska uvažovat jako, v současné době často připomínaný termín, úzký klient, kdy v okně terminálu v podstatě využíváme aplikaci v plné moci serveru, na němž tato aplikace běží. Tento způsob řešení předkládání programů uživateli má spoustu výhod, ale také několik podstatných nevýhod.

Hlavní nevýhodou je především nutná potřeba připojení k serveru, na němž běží námi žádaná aplikace. Další nevýhodou je úroveň soukromí, kdy musí uživatel důvěřovat správci serveru, na němž je jeho aplikace provozována. Omezená je též možnost náročných multimediálních aplikací. Ale nyní k výhodám. Za hlavní výhodu se dá považovat, že uživatel nemá nainstalované aplikace na svém počítači, pouze se terminálem přihlašuje a používá aplikace na serveru. Současně jeho data jsou taktéž uložena na serveru. Z toho vyplývají další

výhody. Uživatel nemusí svoje data ručně přenášet, nemusí také řešit aktualizace software, případně kompatibilitu programů se svými daty. Na straně serveru je očekávána též péče o bezpečné zálohování dat. [16]

## 1.5 Použitý software

Pro veškeré součásti práce bylo rozhodnuto používat pokud možno pouze volně šiřitelné softwarové prostředky.

Jako operační systém serveru, jenž bude ústředím a celým zázemím pro prostředí softwaru připojovaných klientů, byl jako nejvhodnější variantou zvolen operační systém GNU Linux, konkrétně distribuce Debian. Debian především pro jeho dobrou pověst z hlediska nasazení na serverech. Velmi snadnou rozšiřitelnost pomocí rozsáhlého balíčkovacího systému a celkovou spolehlivost ve stabilní větvi Stable. [17]

Jako hlavní prostředek tvorby softwaru a současně nástroj komunikace s operačním systémem je použit jazyk Bash, což je rozšířením Bourne Shellu, nejčastějšího příkazového interpreteru. [15]

Univerzálnost softwarového řešení bude podpořena využitím databázového stroje MySQL. Hlavním důvodem je široká možnost zpracování dat klientů v databázi. Požadavky a veškerá jejich data budou zpracovávána v položkách tabulek databáze. Stejně tak budou řešena i zpracování výstupů, vstupů a mezičástí ukázkových aplikací, což zvýší možnosti tohoto řešení. [13]

Z praktických důvodů a obtížné manipulace s programem vázaným na operační systém, instalovaný databázový stroj a umístěné skripty, je práce vystavěna na virtuálním počítači. Jako vhodný nástroj pro založení virtuálního stroje byl zvolen produkt VMware workstation. Pro jeho omezenou licenční politiku práce pokračovala na programu VMware player s volnou možností použití pro naše účely.

# 2 Úroveň hardware

Problematikou převodu informací mezi počítačem a spotřebičem, případně dílčí částí systému se v současné době zaobírá několik komerčních firem. Z nejakceptovanějších značek v tomto oboru na trhu jmenujme například firmy Sauter, Honeywell, Siemens, Saia, T.A.C.-Schneider a Wago. [10]

Prvky těchto zařízení jsou různé, každá firma se ubírá svojí vlastní vývojovou cestou. Tyto firmy uvádějí na trh jak kompletní řídící podsystém, jenž je opatřen mikrokontrolérem, několika datovými body a některými standardními rozhraními, většinou jako stavebnicová provedení lišící se dle výrobce, kde jádrem stavebnice je řídící jednotka a po sběrnici obsluhuje ostatní jí příslušející zařízení, to jest většinou moduly datových bodů. [12]

## 2.1 Typy datových bodů

### 2.1.1 Digitální výstup

V běžné terminologii označován častěji „Digital output“. Jedná se o jeden z nejčastěji používaných datových bodů. Řídící systém na základě informací ze vstupních datových bodů, případně připojeného terminálu nebo počítače rozhodne o logické úrovni digitálního výstupu. Digitální výstup už podle svého názvu může nabývat dvou různých stavů. Stavu logické nuly odpovídá výstupní úroveň napětí U=0V. Stavu logické jedničky odpovídá úroveň napětí U=24V stejnosměrné, což je v průmyslu velmi běžná hodnota malého napětí. S výhodou se používá například pro spínání relé, rozsvěcování kontrolek nebo ovládání spotřebičů, jenž mají také často rozhraní svého ovládacího vstupu 24V. Proudová zatížitelnost digitálního výstupu je různá v závislosti na osazení modulů různých výrobců, avšak pro ovládací účely, to je užití v ovládacích částech zařízení obvykle postačuje. Někteří výrobci využívají pro své digitální výstupy miniaturních relé, kde je proudová zatížitelnost dána především parametry spínacího kontaktu tohoto relé. Druhou možností je osazení digitálních výstupů spínacími tranzistory. Zde je bází tranzistoru ovládána úroveň napětí na výstupu. Výrobci jako je například Wago zaujímají flexibilní postoj a nabízejí souběžně obě řešení, kdy v rámci podsystému můžeme nasadit jak tranzistorové, tak releové digitální výstupy souběžně a podle předem daných potřeb uplatnit vhodnější z možností. Existují dvě alternativy napájení. První z nich je, že řídící podsystém je opatřen externím napájecím zdrojem a usměrňovačem, napětí

digitálních výstupů je pak vázáno, v případě releových výstupů přímo dáno napětím tohoto externího zdroje. Další možností je vnitřní napájecí zdroj, jako součást jednotky. Toho se využívá především u menších podsystémů s tranzistorovými výstupy.

Jak je uvedeno výše, z logiky digitálních výstupů není nikterak podstatná jejich proudová zatížitelnost. Ze dvou důvodů je možné digitální výstup posílit pomocí externího relé. Tím prvním důvodem je proudové posílení, kdy proudovou zatížitelnost nyní udává kontakt tohoto externího relé. Druhým důvodem je galvanické oddělení spouštěného zařízení. Při galvanickém oddělení se nemusíme obávat, že chyba v zařízení, které ovládáme může zničit digitální výstup, případně celý řídící podsystém, což v závislosti na velikosti celého podsystému a na vzniklém rozsahu poškození může nabývat škody řádově jednotek až stovek tisíc korun českých. [10]

### 2.1.2 Digitální vstup

Digitální vstup je častěji označován jako „Digital input“. Jde o často využívaný datový bod řídících podsystémů. Můžeme jej chápat jako opak digitálního výstupu. Jde o datový bod, jenž očekává na svém vstupu nějakou úroveň napětí a podle ní se rozhoduje o logické jedničce nebo logické nule, kterou předá řídícímu podsystému. Tento vstup se využívá čistě číslicově, proto nemusíme ani řešit, při kterých úrovních napětí dochází k rozhodnutí na logickou hodnotu nula, ve kterých úrovních napětí na logickou hodnotu jedna, případně zda existuje takzvané zakázané pásmo, ve kterém datový bod neumí spolehlivě rozpoznat, které logické hodnoty má právě nabývat.

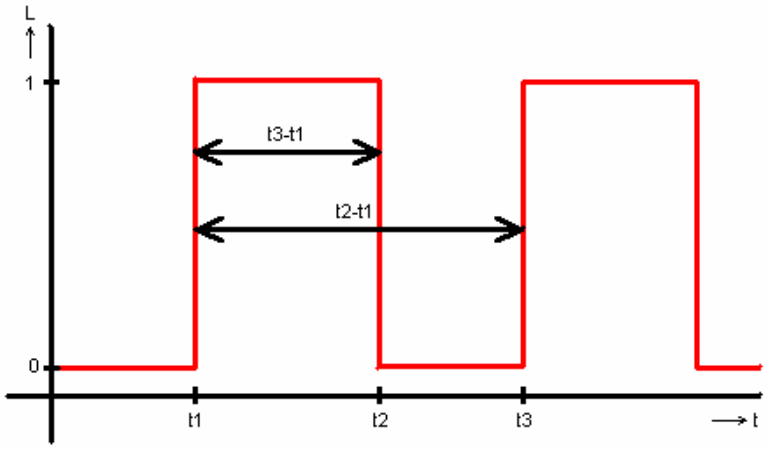
Pro sepnutí na logickou hodnotu jedna je požadovaná úroveň napětí U=24V. Pro logickou hodnotu nula je požadováno napětí U=0V. [10]

I digitální vstup je možno podobně jako digitální výstup galvanicky oddělit od zařízení, z nějž je logická hodnota snímána. A to za pomoci relé, kdy napětím ovládáme relé a datový bod potom napájíme z vlastního zdroje přes spínací kontakt tohoto relé. Této možnosti se však využívá málokdy. Snímaná zařízení totiž často nevysílají hlášku o stavu přímo, avšak očekávají napájení kontaktu, relé umístěného v samotném zařízení a snímán je stav sepnutí právě kontaktu tohoto relé. Z tohoto hlediska se potom zničení řídícího podsystému nějakým parazitním napětím případně výbojem nemusíme obávat.

### 2.1.3 Analogový výstup

Dalším možným datovým bodem je analogový výstup, častěji nazývaný jako „Analog output“. Tento datový bod, jak už název napovídá, řeší ovládací úrovně ve spojitém pásmu. Situace, kdy potřebujeme ovládat nějaké zařízení spojitě. Veškerá zařízení v průmyslu, očekávající řízení napětím obvykle vyžadují rozsah řídícího napětí U=0-10V . V některých ojedinělých případech je z technických důvodů rozsah napětí U=1,6-10V , kde je většině případů nutno stupnici softwarově kalibrovat. Každý z datových bodů analogového výstupu vychází mnohonásobně dráž, nežli datový bod výstupu digitálního. Ovšem v některých situacích je jeho použití nezbytné. Jeho využití je typické pro řízení například servopohonů, kdy nás nezajímají pouze krajní pozice, ale potřebujeme servopohon nastavit na jakoukoliv hodnotu jeho rozmezí. [10]

V některých případech se v rámci úspory analogových výstupů dají využít finty, jak potřebu řízení analogovým výstupem ušetřit a využít pouze výstupu digitálního. První z nich se týká právě řízení servopohonů, jenž vyžadují plynulé řízení. Servopohon je pak nutné dovybavit potenciometrem s lineární charakteristikou, jenž se otáčí stejně se stavem servopohonu. Tento potenciometr potom využijeme jako dělič napětí. Dělič napětí napájíme napětím U=10V stejnosměrných a z jeho jezdce snímáme napětí. Napětí na jezdci poté odpovídá stavu otočení servopohonu. Takto dovybavený servopohon pouze spínáme digitálním výstupem a odečítáme jeho stav analogovým vstupem, který je vysvětlen níže.




**Obr. 1: Obcházení analogového řešení**

Další možnou fintou pro úsporu analogového výstupu se týká především termopohonů. To jsou zařízení pracující na principu roztažnosti kapalin. Velmi často se v současnosti používají například pro otevírání radiátorů ústředního topení. Takovýto ventil nejen, že dokážeme digitálním výstupem udržet ve stavu vypnuto nebo zapnuto, ale přímo jeho stav řídit ve spojitém pásmu časově, zvolením periody a jejího úseku logické hodnoty nula a úseku logické hodnoty jedna. [9]

### 2.1.4 Univerzální analogový vstup

Univerzální analogový vstup označovaný jako „Analog input“ je hojně užívaným datovým bodem. Slovo univerzální v názvu má svůj význam. Analogový vstup je totiž od ostatních typů datových bodů složitější ve smyslu své vstupní charakteristiky. Další rozličností jsou snímané údaje z elektrického hlediska. Pokud snímáme napětí, tak toto napětí je vyhodnocováno v rozmezí 0 až 100% řídícím podsystémem pro napětí U=0-10V . Nebo pokud snímáme proud datovým bodem I=0-20mA taktéž v rozmezí 0 až 100%. V závislosti na výrobci řídícího podsystému se na měření proudu a měření napětí může nahlížet stejně a to tak, že měření napětí je totéž, jako měření proudu, avšak s využitím vnitřní impedance 500Ω. Využije se poté pouze jiná svorka datového bodu. Další nejčastěji používanou možností univerzálního analogového vstupu je snímání impedance. V některých případech hledáme impedanci jako lineární hodnotu. Většinou však potřebujeme snímat například teploty v prostředí, nebo teploty v zařízeních. Pro měření teploty v oboru elektrotechniky se zdárně využívá termistorů. Termistory, jenž používáme musí nabývat kritérií, tak aby mohly být využity pro snímání pomocí analogového vstupu. Většina výrobců už pro své teplotní vstupy nastavuje pevně svoji charakteristiku. Standardními charakteristikami využívanými v průmyslových oblastech a podporovány výrobci řídících podsystémů jsou nejčastěji Ni1000, někdy Pt1000 méně častěji Ni10000 nebo Pt100. [8]

# 3 Popis paralelismu

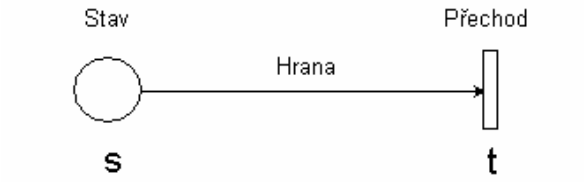
Paralelismem se zabývá například teorie Petriho sítí, jedná se o grafické znázornění míst a přechodů, na jejichž základě můžeme modelovat vytvářený program. Od základu se tedy vyhýbáme situacím kolize, kdy se projeví slabé místo programu, tedy výjimka, se kterou nebylo počítáno při návrhu.

Teorie Petriho sítí vznikla v roce 1962. Poprvé byla použita v doktorské práci C. A. Petriho. Jde o koncept popisu vzájemné závislosti mezi podmínkami a událostmi modelovaného systému. Využívají se pro návrh a popis paralelních architektur. V současnosti především v počítačové oblasti paralelních databázových systémů a komunikačních protokolů v počítačových sítích. Avšak také v telekomunikacích, automatizovaných průmyslových zařízeních a podobně. [7]

Na Petriho sítě lze nahlížet jako na zobecnění konečných automatů. Konečné automaty definují svoje chování posloupnostmi událostí a současně změnami stavu. Konečný automat je jednoznačně definován množinou stavů, přičemž každý stav je jiný a další stavy jsou nemožné. Rozdíl tedy spočívá v tom, že Petriho sítě jsou od konečných automatů definovány svými parciálními stavy v jednotlivých místech sítě. Stavy v místech budeme dále značit množstvím tokenů. [7]

## 3.1 Koncepce

Potřebou tohoto projektu jsou základní vlastnosti a činnosti Petriho sítí. Obeznamme se nyní se značkami, jimiž jsou prvky této sítě vždy reprezentovány viz. obr.2. Jde o stavy, přechody a hrany. Stavy jsou graficky znázorněny jako malé kružnice. Přechody zase jako malé obdélníky, případně silnější úsečky, v tomto dokumentu se však bude dodržovat jedno grafické značení obdélníky. Hrany jsou šipky spojující vzájemně stavy a přechody.



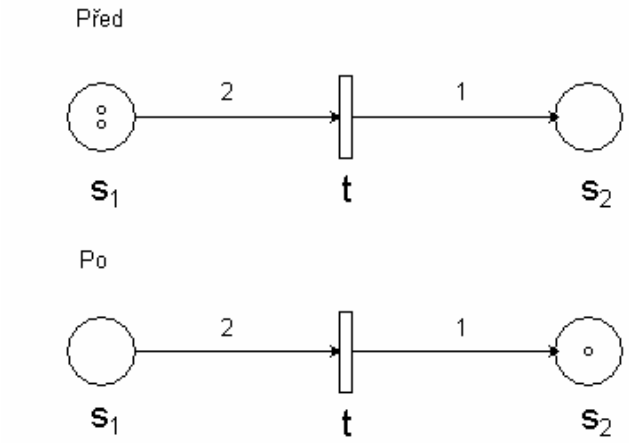

**Obr. 2: Základní součásti grafů**

V teoretickém nadhledu můžeme pohlížet na místa Petriho sítě jako na podmínky. Na přechody zase jako na události. Přičemž hrany znázorňují spojení podmínek s událostmi a naopak událostí s podmínkami. viz. [11]

Místa Petriho sítě můžou nabývat hodnot, graficky označovaných jako tečky uvnitř kružnice. Tyto tečky označují tokeny příslušející danému místu. Podmínka místa bude splněna, pokud místo obsahuje dostatečné množství tokenů. To znamená tolik, aby počet tokenů byl stejný nebo větší, než počet vycházejících hran. Přechod zareaguje, pokud všechny hrany do něj vstupující předají svůj token. Při splnění podmínky jsou tokeny předány přechodu, z místa se odečtou. Existují také násobné hrany, jde pouze o grafické znázornění více jednoduchých hran, jenž začínají ve stejném počátku a končí ve stejném cíli. Poznáme je podle číselného označení, které vyjadřuje pouze násobek hrany. Obě možná vyjádření násobné hrany jsou na obr. 3.


**Obr. 3: Možnosti zobrazení dvojnásobné hrany**

Jednoduchý příklad předání tokenů je na obr. 4. První stav vlastní dva tokeny, protože je splněna podmínka, jsou předány po hraně přechodu, jenž nemá důvod nezareagovat a vyšle token svojí výstupní hranou.

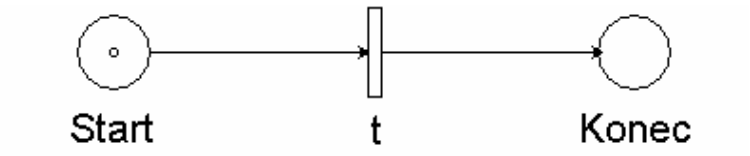



**Obr. 4: Příklad předávání tokenů**

## 3.2 Podstata

### 3.2.1 Cyklický proces

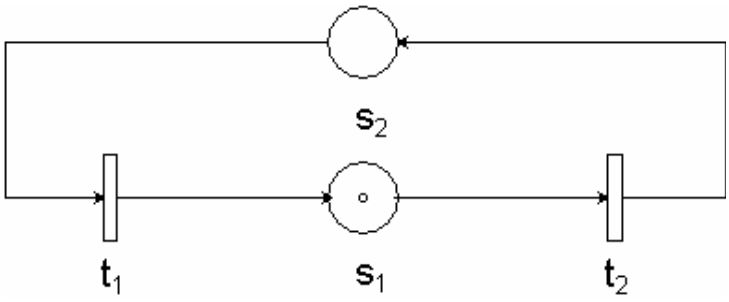
Velkým rozšířením možností Petriho sítí je užití tzv. cyklického procesu. Dosud jsem se zmínil pouze o procesech se začátkem a koncem, jenž v prvopočátku nějakým impulsem začnou a poté v nějakém stavu skončí. Na obr. 5 je ukázka nejjednoduššího příkladu.




**Obr. 5: Jednoduchý necyklický proces**

Na začátku tohoto příkladu z obr.4 existuje místo Start s jedním tokenem, tím se spustí proces. V tomto případě ihned skončí s výsledkem držení jednoho tokenu koncovým místem.

Dalším příkladem bude samotná podstata problému cyklického procesu. Na obr. 6 je jednoduchý cyklus, jenž není přímo odstartován spouštěcím místem.




**Obr. 6: Jednoduchý cyklický proces**

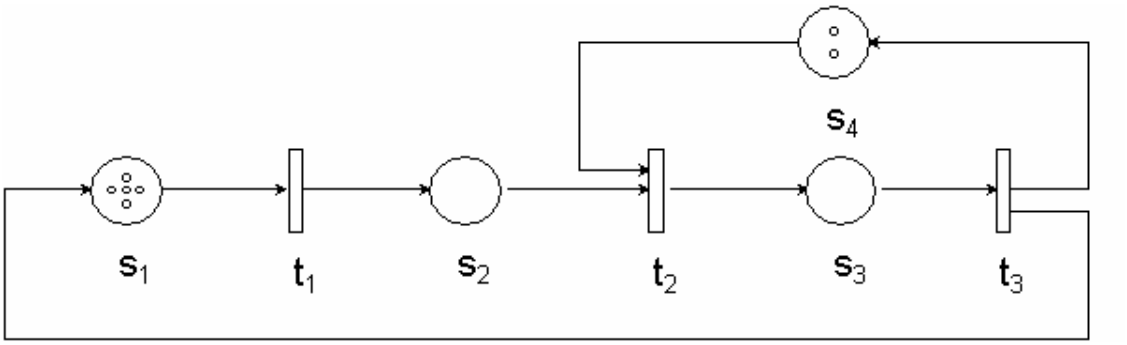
V tomto případě bude token neustále předáván do nekonečna. Není zde třeba vnějšího zásahu, jakým bylo místo Start.

### 3.2.2 Sdílení zařízení

Žádný proces však nevzniká samovolně. Vznik procesu má svůj účel a přesný řád. Petriho síť tedy vychází z naší potřeby obsluhovat jedno zařízení více procesy, kdy situací ve stavu míst jednotlivých procesů dokážeme eliminovat souběžný přístup k jednomu

zařízení. Toto je účelem využití Petriho sítí v našem problému. Systém je bezpečný, pokud více procesů nemůže současně zapisovat do jednoho zařízení, nebo číst neúplné údaje ze zařízení, případně číst v nevhodnou dobu.

Možným příkladem sdílení zařízení více procesy je situace na obr. 7. Konkrétně jde o situaci pěti procesů sdílejících dvě zařízení.




**Obr. 7: Sdílení zařízení více procesy [11]**

Místo s1 vyznačuje počet zařízení nevyužívajících zařízení. Přechod t1 označuje vznik požadavku na přidělení zařízení. Místo s2 znázorňuje počet procesů čekajících na zařízení. Přechod t2 zaručuje přidělení zařízení procesu. Místo s3 udává počet procesů využívajících zařízení. Přechod t3 je uvolnění zařízení procesem. Zbývající místo s4 označuje nevyužitá zařízení. viz. [11]

### 3.2.3 Diskrétní pojetí času

Jednou z praktických vlastností tohoto řešení je též pojetí času, kdy při počítačové simulaci nemusíme brát zřetel na skutečný čas. Ten se odvíjí od abstraktních jednotek, čili kroků. Kroky jsou do procesu zanášeny podle modelu na obr. 4. To je místem, jehož stav je nezávisle řízen z vně procesu.

Pro nás v prvopočátku není důležité, jak je jeden krok dlouhý v jednotce času. Nemusí nás tedy obtěžovat ani výkonnost počítače, na kterém simulaci testujeme. Při správné synchronizaci vláken a bezchybném sdílení zařízení procesy dopadne výsledná charakteristika na každém počítači stejně.

# 4 Realizace databáze

MySQL je open-source databázový systém založený na tabulkách. Řádky zastupují jednotlivé záznamy, ve sloupcích jsou pak obsaženy jednotlivé hodnoty. Tento způsob zápisu je ideální pro různé seznamy, ceníky, zkrátka jakákoli data zapisovatelná pomocí tabulky.

Velkou výhodou MySQL je rychlost a jednoduchost. Běží totiž, jako samostatný server a k práci s daty se většinou využívá skriptování na straně serveru (např. pomocí PHP). I proto našel velké využití právě na internetu. [15]

Vyjma MySQL existují ještě jiné databázové systémy založené na SQL, tedy na standardním dotazovacím jazyce pro práci s databázemi, například PostgreSQL, nebo MSSQL.

S MySQL se pracuje pomocí příkazů, které se zakončují středníkem. Základním prvkem MySQL je databáze. Ta obsahuje tabulky, které mají definovaný počet sloupců. Do řádků v tabulkách se pak ukládají samotná data. Ta mohou mít různý formát, který ale musí odpovídat parametrům daného sloupce. [15]

## 4.1 Spuštění serveru

Jako serverový operační systém je zvolen Debian 5 „Lenny“ z větve Stable, protože jde o spolehlivý operační systém, u kterého nejsou předpokládány závady, které by negativně ovlivňovaly funkčnost projektu.

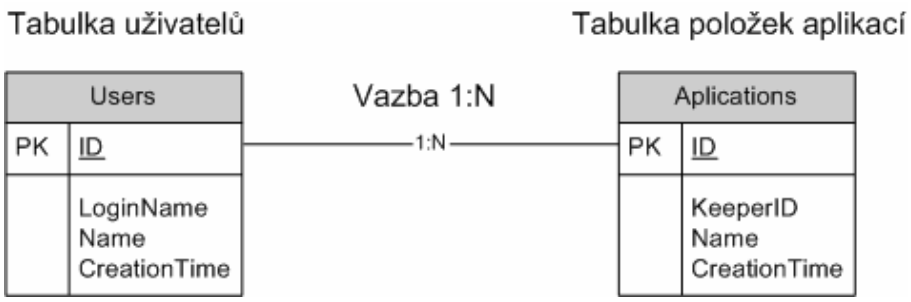
Pokud spouštíme službu mysqld, je třeba být přihlášen jako administrátor operačního systému „root“ a zadat příkaz: safe_mysqld & . My ovšem předpokládáme automatické spuštění databázového serveru automaticky po startu systému.

Podstatným předpokladem je spuštěná služba sshd, jež je nezbytná pro připojení šifrovaného terminálu.

## 4.2 Uživatelé a položky aplikací

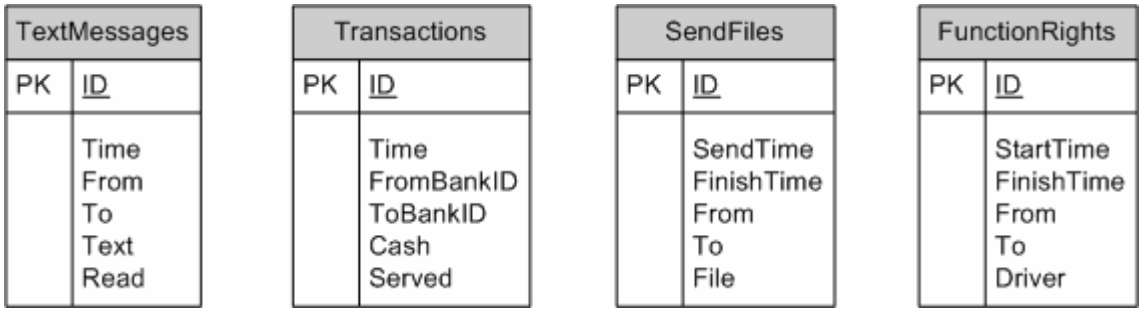
Kromě založení uživatelů programu jako účtů unixového systému je třeba též jejich zapsání do tabulky uživatelů „Users“. Databáze potom pracuje s údaji v této tabulce ve vazbě na tabulku aplikací „Aplications“, ve které je soupis všech vytvořených aplikací. Uživatel

může být klientem jedné nebo i více aplikací z těchto položek, jak naznačuje vazba 1:N mezi těmito tabulkami.




**Obr. 8: E-R diagram vztahu uživatelé – položky aplikací**

Aplikace samy o sobě jsou ztvárněny vlastními tabulkami, v nichž jsou deklarovány podstatné položky pro jejich správnou funkčnost. Na obr. 9 jsou znázorněny ukázkové aplikace s jejich položkami.

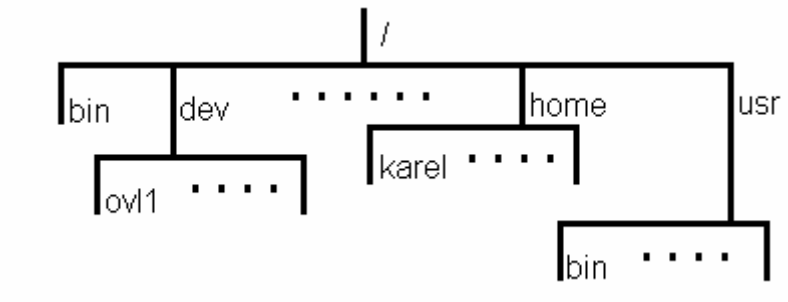



**Obr. 9: Tabulky ukázkových aplikací**

Toto jsou ukázkové možnosti řešení, podle něhož je program možno dále rozvíjet. Samotné aplikace jsou navrženy s co největší podobností tak, aby návrh dalších aplikací nevyžadoval pokud možno žádný administrativní zásah do databáze nebo do operačního systému.

# 5 Tvorba aplikací

Tento ukázkový program je vzorovým řešením a možným příkladem využití tohoto projektu. Je zde prezentováno účelové využití databáze a všech potřebných řešení, jež v principu projektu využíváme, včetně ošetření přístupovými právy, což je nezbytnou součástí a základem jádra bezpečnosti celého principu.

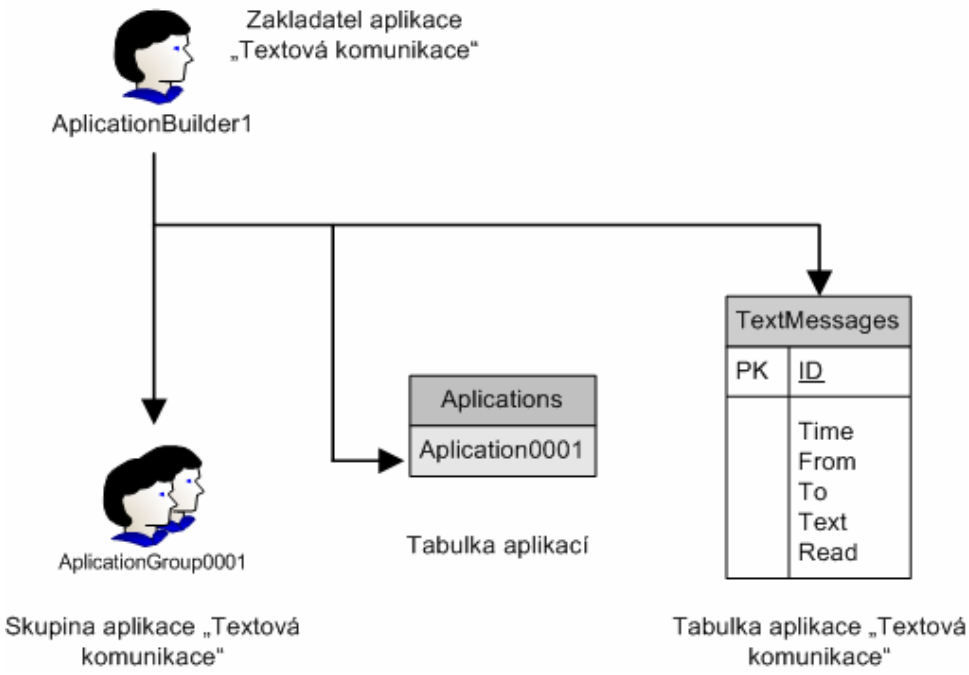



**Obr. 10: Podstatné adresáře projektu**

## 5.1 Textová komunikace

Textová komunikace mezi klienty je jedním z nejvýstižnějších příkladů využití komunikačního softwaru. Jsou zde předvedeny vlastnosti použitelné i ve složitějších aplikacích, avšak nevyužívá plně jeho možností. Těm jsou věnovány další aplikace, pro něž jsou vlastnosti komunikačního softwaru charakterističtější a zřetelněji je na nich znát jejich význam užití.

Člen skupiny „AplicationBuilders“ založí novou skupinu. V našem případě je pojmenována „AplicationGroup0001“. Do této skupiny bude později možno registrovat členy. Členem se rozumí uživatel aplikace. V dalším kroku je třeba v tabulce databáze „Aplikace“ vytvořit záznam o vzniku aplikace. Prakticky to znamená vytvoření řádku. V řádku se vyplňují položky „ID“, v tomto případě „0001“; název aplikace jako „TextovaKomunikace“; a nakonec majitel aplikace. Majitelem aplikace je „AplicationBuilder1“, v jehož domovském adresáři nalezneme také obslužný skript shodný s názvem aplikace, tedy „TextovaKomunikace.sh“.




**Obr. 11: Založení aplikace textové komunikace**

Prvotní službou tohoto programu je jednoduchá textová komunikace různých klientů obslužného serveru, jenž používají vzdálených terminálů k připojení. Jedná se tedy o jeden z podprogramů. Princip komunikace probíhá následovně: Klient vybere příkaz napsat. Poté zadá identifikační číslo, případně přihlašovací jméno klienta, kterému chce zprávu odeslat, tedy adresáta. Posléze napíše vlastní text. Odesláním se údaje zapíší do tabulky „Rozhovor“. Odesilateli se také vrátí odeslaná zpráva zpět na terminál jako důkaz zápisu v databázi a též pro zpětnou kontrolu odeslaného textu během rozhovoru. Spolu s položkou je také zobrazen symbol, jako doručenka o přijetí. Odesilatel má také možnost vidět uživatele služby proto, aby mohl rozpoznat, komu psát může a komu nikoliv.

```
mysql> select * from TextMessages;


+----+---------------------+------+----+--------------+------+
| ID |        Time         | From | To |     Text     | Read |
+----+---------------------+------+----+--------------+------+
|  1 | 2009-03-29 14:35:36 |    1 |  2 | Pokusny text |    1 |
+----+---------------------+------+----+--------------+------+
|  1 | 2009-03-29 14:55:22 |    2 |  1 | Odpoved      |    1 |
+----+---------------------+------+----+--------------+------+
|  1 | 2009-03-29 14:58:26 |    1 |  2 | Ahoj         |    1 |
+----+---------------------+------+----+--------------+------+
|  1 | 2009-03-29 15:02:59 |    1 |  2 | Ahoj         |    0 |
+----+---------------------+------+----+--------------+------+
```

**Obr. 12: Tabulka textové komunikace**

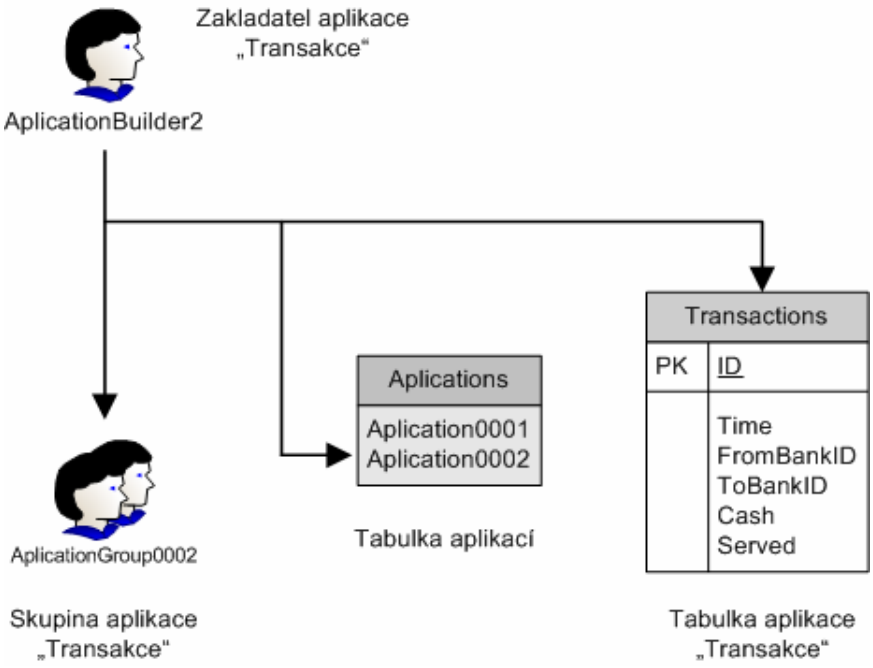
## 5.2 Transakce peněz

Další službou a současně aplikací je možnost transakce peněz. V principu je služba velmi podobná službě textové komunikace. Rozdíl spočívá v odečtu kreditu na účtu klienta a připsání dané sumy na účet adresáta. Opět klient vybere adresáta, určí sumu a odešle. V okně se opět zobrazí doručenka o řádném převodu kreditu. Nebo se zobrazí hláška o jednoznačné stornaci převodu a důvod nepřevedení peněz na jiný účet. Důvodem může být nedostatek finančního kreditu odesílatele, neexistence účtu adresáta, chyba systému během vykonávání, nebo nějaký jiný možný důvod.

```
mysql> select * from Transactions;


+----+---------------------+------+----+------+--------+
| ID |        Time         | From | To | Cash | Served |
+----+---------------------+------+----+------+--------+
|  1 | 2009-04-11 16:35:32 |    1 |  2 |  100 |      1 |
+----+---------------------+------+----+------+--------+
|  1 | 2009-04-11 16:52:22 |    1 |  3 |  200 |      2 |
+----+---------------------+------+----+------+--------+
|  1 | 2009-04-11 16:48:46 |    3 |  1 | 1000 |      1 |
+----+---------------------+------+----+------+--------+
```

**Obr. 13: Tabulka transakcí**




**Obr.14: Založení aplikace transakcí**

## 5.3 Virtuální centrální banka

Pojmem virtuální centrální banky serveru rozumějme správu serveru nad financemi uživatelů. Mějme na paměti možnost, kdy finanční kredity uživatelů mohou vycházet z nákupu penězi z reálného světa a tyto finanční kredity posléze nepodléhají žádnému krytí. Bez žádné kontroly je tedy kurz finančního kreditu virtuální měny pouze v rukou provozovatele serveru. Bez řádné záruky peněz virtuálního světa jsou tedy poplatné služby v rámci serveru nedůvěryhodné a vhodné jen jako hračka bez záruk. Existuje více možných řešení této problematiky. Nejjednodušším způsobem je stanovení kurzu podle již existující reálné měny, například Koruny české. V takovém případě virtuální centrální banka pouze zprostředkovává převod peněz ze světa skutečného do virtuální podoby prostředí serveru. A také vyplácení peněz z virtuálního prostředí do skutečného světa.

Další poněkud sofistikovanější možností je nákup podílnictví serveru. To je situace, kdy celková suma veškerého virtuálního kreditu odpovídá hodnotě v reálném světě. Řešením může být část, nebo i celkové jmění společnosti vlastnící server. Nákupem kreditů virtuálního

světa se poté stává klient současně i skutečným podílníkem reálné společnosti, což opět zamezuje podvodům a neregulérnímu jednání.

Nejméně šťastným řešením, jestli se to dá vůbec nazývat řešením, do dnešní doby však jediné realizované ve světě počítačových online her, principiálně však nevhodné pro vážné použití, je nákup kreditu bez jakýchkoliv záruk. Nákupem reálnými penězi získá klient určitý kredit, avšak nemůže předpokládat jeho stabilitu. Spekulativním jednáním může provozovatel serveru bezmezně prodávat virtuální hodnoty, případně nekontrolovaně devalvovat virtuální měnu za účelem dalšího snadného prodeje.

Úkolem centrální banky je tedy především krytí finančního kreditu, tedy kontrola nad dodržováním vzniku financí, případně jejich zániku. A také kontrola nad sumou veškerého finančního kreditu v oběhu programu.

Z praktického hlediska je to tak, že při každém transakčním pohybu je kredit odesílajícího účtu předán centrální bance a ta kredit odevzdá účtu cílovému. Je tak zamezeno podvádění ze strany správců bank.

## 5.4 Vedení finančních účtů

K zajištění konkurence vedení finančních účtů není vhodné, aby kreditní prostředky ležely během svého nevyužití v rukou virtuální centrální banky. Za tímto účelem je vhodné vytvořit prostor a umožnit tak vznik komerčních bank.

Banky jsou realizovány jako účast klienta k určité službě. V pojmu databáze potom klient banky je v tabulce uživatelů dané banky a v tabulce transakcí může účtem v této bance disponovat.

Zakládat banky mají opět právo členové skupiny „AplicationBuilders“, protože se jedná o zásah ve smyslu založení aplikace. Takovéto uživatele nazývejme dále jako správce bank.

# 6 Řešení práv funkcí

Velmi běžnou činností, již očekáváme od komunikačních prostředků v dnešní době je posílání souborů. I tato činnost není opomenuta v řešení komunikačního softwaru. Ba naopak, je zde využito dalších možností, využívajících vlastností unixového serveru. A to jednotného přístupu k zařízení jako k běžnému souboru. Využitím tohoto faktu je již nastíněna myšlenka dalekosáhlosti tohoto postřehu.

Řešení vychází z přirozené vlastnosti systému a tou je možnost vytváření skupin uživatelů. Skupiny uživatelů jsou z tohoto hlediska mocným prostředkem pro spojení uživatelů, kteří oplývají některou společnou vlastností. Na této vlastnosti je založen princip zasílání souboru a také udělování práv pro použití funkcí.

Uživatelé se běžně ukládají jako položky do konfiguračního souboru /etc/passwd. Velmi obdobným způsobem je taktéž řešena účast v uživatelských skupinách.

| 1 | : | 2 | : | 3 | : | 4 | : | 5 | : | 6 | : | 7 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| karel | | x | | 1000 | | 1000 | | karel | | /home/karel | | /bin/bash |

**Obr.15: Struktura uložení uživatele v operačním systému**

Na obr. 15 je znázorněna struktura a příklad zápisu v souboru /etc/passwd. Jednotlivá čísla znamenají:

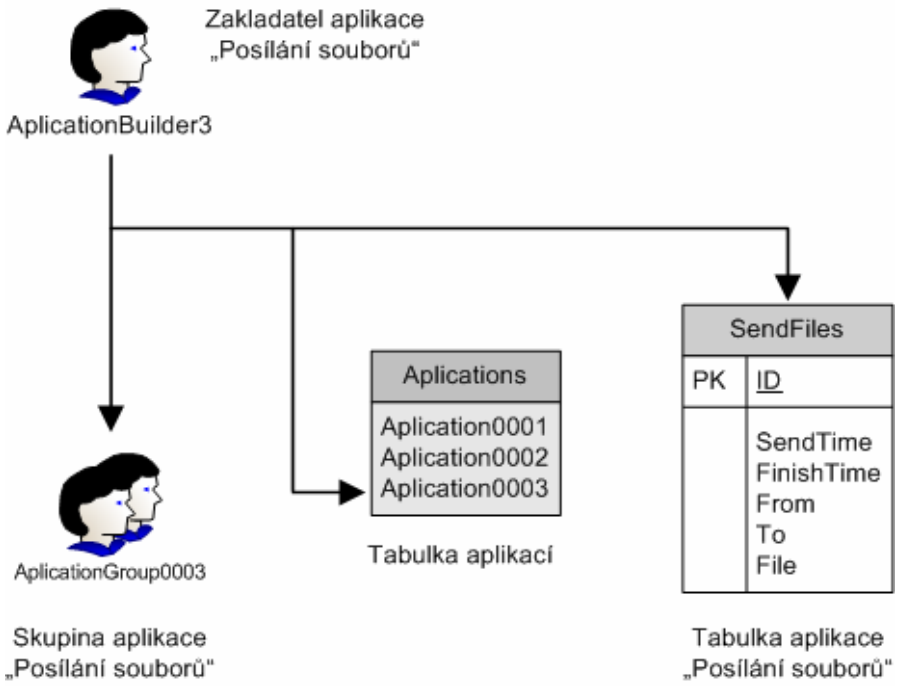
1. Uživatelské jméno
2. Heslo
3. Identifikační číslo uživatele
4. Hlavní skupina uživatele
5. Informace o uživateli
6. Domovský adresář
7. Upřednostňovaný příkazový interpret

Dle těchto položek je v unixových operačních systémech jmenován výčet vlastností uživatele tak, jak na něj nahlíží operační systém. Další podrobná nastavení najdeme na jiných místech, jako například ve skrytých souborech domovského adresáře uživatele.

## 6.1 Posílání souboru

Posílání souboru probíhá následujícím způsobem. Klient odešle požadavek pro odesílání souboru. Tím prakticky založí jednoúčelovou skupinu uživatelů. Do této skupiny zařadí sám sebe. Poté do ní zařadí adresáta, jemuž soubor zasílá.Tato skupina má nastavenu dobu svojí životnosti, to je času, do nějž se předpokládá potřebnost poskytování souboru. Během delšího času používání programu by tabulka databáze posílaných souborů ztratila na přehlednosti. Mohlo by se též stát, že zasilatel souboru by daný soubor odstranil a pod stejným názvem založil soubor jiný, u kteréhož by jeho poskytnutí nepředpokládal.

Přístupová práva souboru se samozřejmě nastaví na čtení, případně spuštění pro zmíněnou skupinu. Nezapomeňme brát v potaz, že příjemce i adresát jsou uživatelé stejného počítače. Nyní adresát může dle libosti a přístupových práv svůj přijímaný soubor zkopírovat pro svoji potřebu, případně spustit.




**Obr.16: Založení aplikace posílání souborů**

Aplikace posílání dat je trošku rozdílná od předchozích modelů toho, jak byly předchozí aplikace prezentovány. Je zde patrnější provázanost s využitím vlastnických práv uvnitř skupin.

Člen skupiny „AplicationBuilders“ založí novou skupinu. V našem případě je pojmenována „AplicationGroup0003“. Do této skupiny bude později možno registrovat členy. Členem se rozumí uživatel aplikace. V dalším kroce je třeba v tabulce databáze „Aplications“ vytvořit záznam o vzniku aplikace. Prakticky to znamená vytvoření řádku. V řádku se vyplňují položky „ID“, v tomto případě „0003“; název aplikace jako „TextovaKomunikace“; a nakonec majitel aplikace. Majitelem aplikace je „AplicationBuilder3“, v jehož domovském adresáři nalezneme také obslužný skript shodný s názvem aplikace, tedy „PosilaniSouboru.sh“.

```
mysql> select * from SendFiles;


+----+---------------------+---------------------+------+----+---------+
| ID |       SendTime      |      FinishTime     | From | To |  File   |
+----+---------------------+---------------------+------+----+---------+
|  1 | 2009-04-15 16:35:32 | 2009-04-16 00:00:00 |    1 |  2 | soubor1 |
+----+---------------------+---------------------+------+----+---------+
|  2 | 2009-04-15 18:02:09 | 2010-12-31 00:00:00 |    1 |  2 | soubor2 |
+----+---------------------+---------------------+------+----+---------+
```

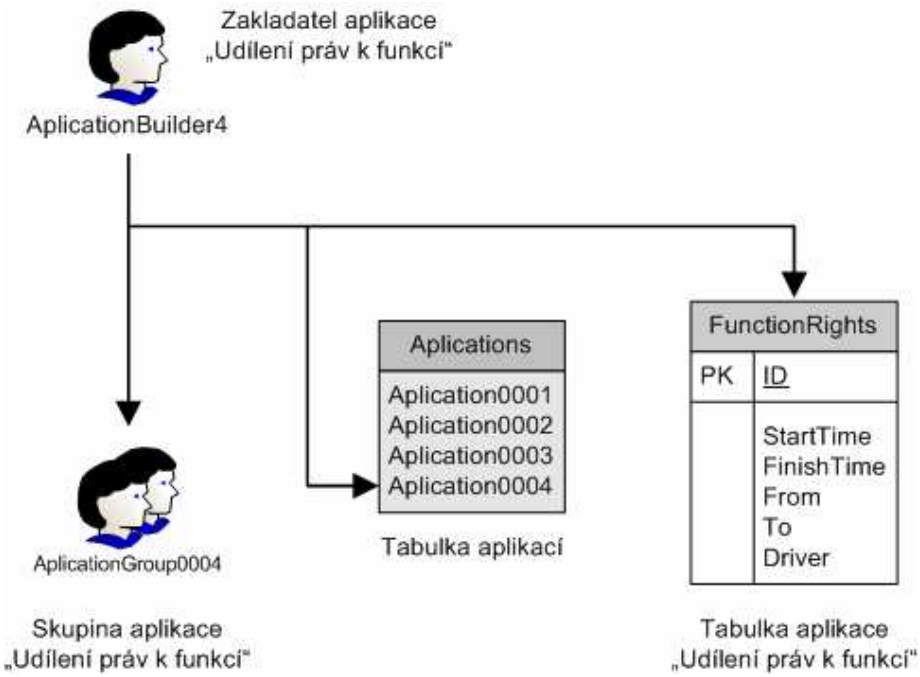
**Obr. 17: Tabulka posílání souborů**

## 6.2 Udělení práva k funkci

### 6.2.1 Založení aplikace

Podobně, jako může klient klientu poslat soubor, může udělit i libovolné právo na funkci, na níž má sám právo.

Udílení práv funkcí je velmi obdobné aplikaci „Posílání dat“. Přestože se jedná v principu úplně o jiné využití, je zde využita vlastnost Unixu, toho, že k ovladačům zařízení je možno přistupovat stejně jako k jakémukoliv souboru. Rozdíl spočívá především v tom, že není řešena doba vypršení práva k funkci tak, jak tomu bylo u posílání souboru.




**Obr. 18: Založení aplikace udílení práv funkcí**

```
mysql> select * from FunctionRights;


+----+---------------------+---------------------+------+----+--------+
| ID |       SendTime      |      FinishTime     | From | To | Driver |
+----+---------------------+---------------------+------+----+--------+
|  1 | 2009-04-15 16:35:32 | 2009-04-16 00:00:00 |    1 |  2 | testDr |
+----+---------------------+---------------------+------+----+--------+
|  2 | 2009-04-15 18:02:09 | 2010-12-31 00:00:00 |    1 |  3 | testDr |
+----+---------------------+---------------------+------+----+--------+
```

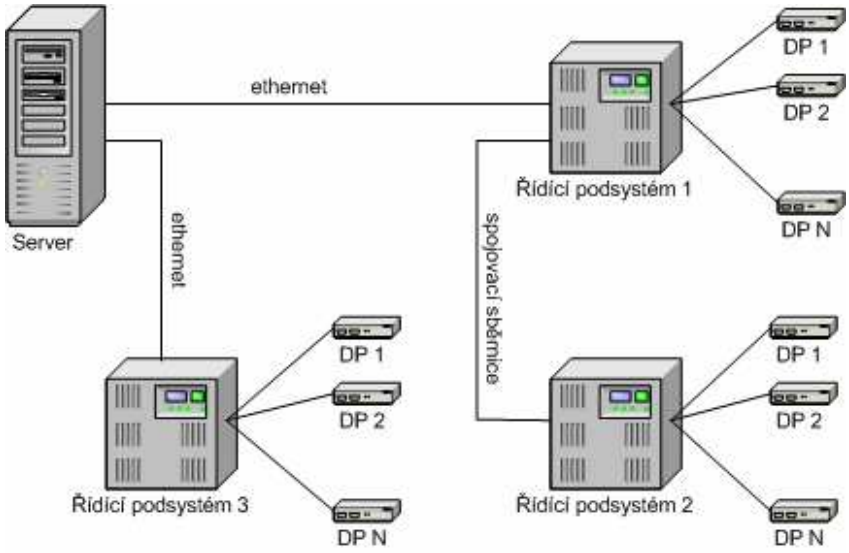
**Obr. 19: Tabulka udílení práv funkcí**

### 6.2.2 Možnosti aplikace

Aplikace funguje na principu stanovování a řešení práv jednotlivých uživatelů pro přístup k ovladačům. Ovladače jsou v unixových systémech řešeny jako běžné soubory. Proto definujme sadu ovladačů pro veškerá periferní zařízení.

Veškeré ovladače potřebujeme nejen pro hardware zařízení, ale i pro jednotlivé součásti zařízení v řídícím podsystému. Každý datový bod tedy bude mít vlastní ovladač a je možné

s ním dále pracovat jako s nezávislým dílčím prvkem a řešit k němu samostatně přístupová práva.

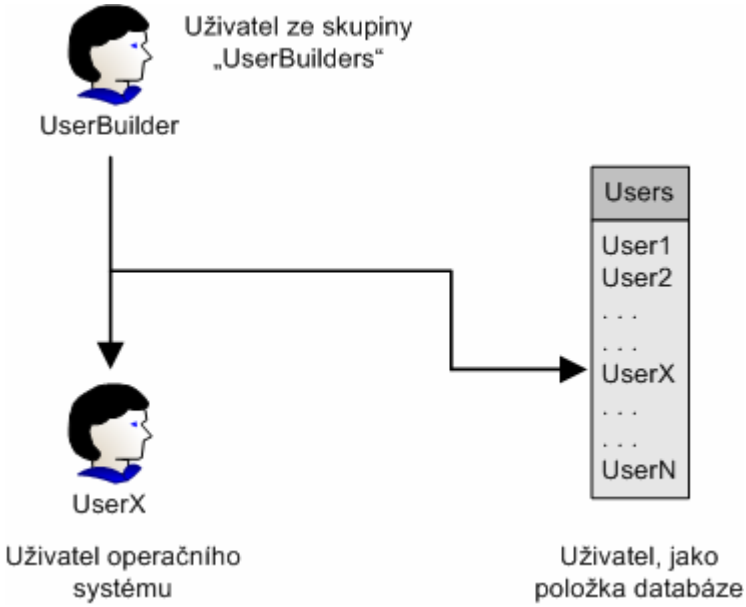



**Obr. 20: Hierarchie řídícího systému**

Tento přístup je zdánlivě nepodstatný do chvíle, kdy považujeme jeden řídící podsystém jako jednu jednotku zařízení a nemusíme členit jeho přístupová práva. Důvody pro dílčí ovladače jsou tedy dva. Prvním z nich je členění práv i v rámci jednoho zařízení nezávisle na jednotce, případně rozložení do více jednotek řídícího podsystému. Uveďme příklad: Jeden z uživatelů může mít právo sledovat teplotu v místnosti, ale nemusí už mít právo tuto teplotu změnit. Druhý uživatel může mít i právo teplotu v místnosti změnit. Druhým podstatnějším důvodem je cena a praktická stránka používání řídících podsystémů. Je velmi pravděpodobné, že z finančních důvodů budou veškerá i spolu nikterak nesouvisející zařízení spojena v jednom řídícím podsystému. V takovém případě by potom nebyla možnost rozlišovat jednotlivá zařízení.

## 6.3 Založení uživatele

K založení uživatele je třeba přistupovat z více hledisek, než pouze jako příkaz pro přidání uživatele v unixovém systému. Jednak jako založení uživatele v systému s určitými právy. Také definování věcí v sounáležitosti komunikačního programu. Ale také založení uživatele jako položku databáze komunikačního programu.




**Obr. 21: Obecné založení uživatele**

```
mysql> select * from Users;


+----+-----------+--------+---------------------+
| ID | LoginName |  Name  |    CreationTime     |
+----+-----------+--------+---------------------+
|  1 | karel     | Karel  | 2009-03-01 15:07:06 |
|  2 | venca     | Vaclav | 2009-03-04 17:22:12 |
|  3 | uzivatel1 | u1     | 2009-03-16 22:01:00 |
|  4 | uzivatel2 | u2     | 2009-03-16 22:08:56 |
|  5 | uzivatel3 | u3     | 2009-03-16 22:25:23 |
|  6 | uzivatel4 | u4     | 2009-03-16 22:27:02 |
+----+-----------+--------+---------------------+
```

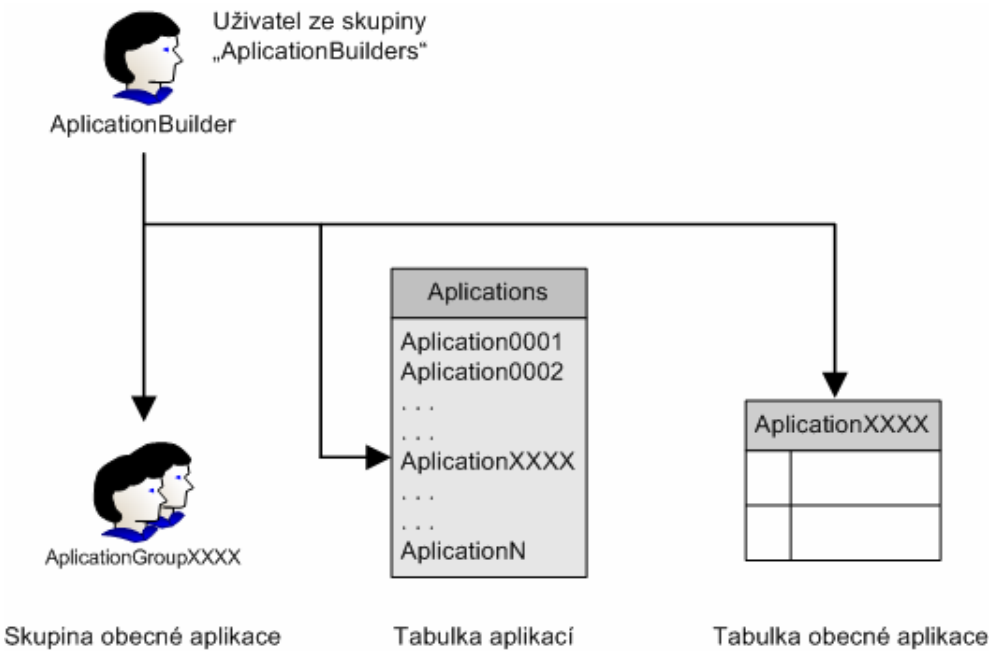
**Obr. 22: Tabulka uživatelů**

## 6.4 Obecné založení aplikace

Založení aplikace komunikačního programu je komplexnější záležitost. Z důvodů maximální kompatibility je mojí snahou veškeré kroky co nejvíce sjednotit, aby samotný proces založení některé z aplikací nevyžadoval pokud možno žádné zásahy nebo změny v komunikačním programu.

Podstatnou informací a důvodem, proč je zde řešeno zakládání aplikace takto rafinovaným způsobem je možnost zakládání a doplňování aplikací za běhu komunikačního programu. Nikoliv tedy založení programu s předem danými službami a jeho následná možnost aktualizace, ale přímo interaktivní možnost doplnění bez ohrožování chodu zbytku služeb. A celkové navázání na již zpřístupněné služby.

Samotné obecné založení sestává z několika hlavních kroků. V konkrétních aplikacích jsou často důležité všechny tyto kroky, avšak v některých jednoduchých příkladech aplikací, jež jsou uvedeny, často nevyužit jeden z kroků.




**Obr. 23: Založení obecné aplikace**

Každá samotná aplikace v komunikačním programu je založena jako uživatel. Je předpokládán účet, jenž má dostatečná oprávnění zakládat další uživatelské účty, zakladatelem musí být člen skupiny „AplicationBuilders“.

Dalším krokem založení je vytvoření skupiny uživatelů, které je pevně vázáno s aplikací. Název této skupiny je zde stavěn jednotně a během vytváření se obměňuje pouze pořadové číslo aplikace. Tvar názvu skupiny je tedy „AplicationGroupXXXX“, kde XXXX je nahrazeno skutečným číslem aplikace. A současně je to i hodnota ID aplikace v položkách databázové tabulky aplikací.

Jak již bylo nastíněno, aplikace jsou úzce spjaty s databází, stejně jako ostatní části komunikačního programu. Pro každou aplikaci je nezbytný zápis do tabulky databáze, s již zmíněným pořadovým číslem. Toto je důležité jak pro jednotnost a kompletnost celého projektu, tak ještě z důvodu řešení práv uživatelů k přístupu k jednotlivým aplikacím.

```
mysql> select * from Aplications;


+----+----------+----------------+---------------------+
| ID | keeperID |      Name      |     CreationTime    |
+----+----------+----------------+---------------------+
|  1 |        3 | TextMessages   | 2009-03-19 14:19:08 |
|  2 |        4 | Transactions   | 2009-03-19 22:12:00 |
|  3 |        5 | SendFiles      | 2009-03-20 15:08:21 |
|  4 |        6 | FunctionRights | 2009-03-22 18:06:45 |
|  5 |        1 | u3             | 2009-04-18 10:19:59 |
|  6 |        1 | u4             | 2009-04-18 10:21:29 |
+----+----------+----------------+---------------------+
```

**Obr. 24: Tabulka aplikací**

Při založení aplikace je též předpokládáno, a ve všech mnou prezentovaných příkladech také využito založení nové tabulky databáze. Tato tabulka je v režii dané aplikace a je čistě využívána pro zpracovávání provozních dat, svojí mateřské aplikace. Jsou zde zaznamenávány veškeré zákroky komunikační aktivity klientů aplikace.

## 6.5 Možnosti aplikace "Udílení práv funkcí"

Aplikace funguje na principu stanovování a řešení práv jednotlivých uživatelů pro přístup k ovladačům. Ovladače jsou v unixových systémech řešeny jako běžné soubory. Proto definujme sadu ovladačů pro veškerá periferní zařízení.

Veškeré ovladače definujme nejen pro hardware zařízení, ale i pro jednotlivé součásti zařízení v řídícím podsystému. Každý datový bod tedy bude mít vlastní ovladač a je možné s ním dále pracovat jako s nezávislým dílčím prvkem a řešit k němu samostatně přístupová práva.

Tento přístup je zdánlivě nepodstatný do chvíle, kdy považujeme jeden řídící podsystém jako jednu jednotku zařízení a nemusíme členit jeho přístupová práva. Důvody pro dílčí ovladače jsou tedy dva. Prvním z nich je členění práv i v rámci jednoho zařízení nezávisle na

jednotce případně rozložení ve více jednotek řídícího podsystému. Uveďme příklad: Jeden z uživatelů může mít právo sledovat teplotu v místnosti, ale nemusí už mít právo tuto teplotu změnit. Druhý uživatel může mít i právo teplotu v místnosti změnit. Druhým podstatnějším důvodem je cena a praktická stránka používání řídících podsystémů. Je velmi pravděpodobné, že z finančních důvodů budou veškerá i spolu nikterak nesouvisející zařízení spojena v jednom řídícím podsystému. V takovém případě by potom nebyla možnost rozlišovat jednotlivá zařízení.

# 7 Poplatnost služeb

V rámci možností řešení finančních služeb komunikačního programu je možné stavět nové aplikace. Jako ukázkový příklad manipulace s finančním kreditem slouží aplikace "Transakce". Možnosti práce s finančním kreditem jsou však mnohem širší. Předpokládejme nasazení v komerční sféře. Máme možnost stavět nové aplikace, které v širším měřítku pracují s kapitálem. To znamená například účtování poplatků za některou službu. Můžeme si pod tím představit jednorázový poplatek za vykonání některého příkazu. Nebo také poplatek za zpřístupnění některé služby. Nejsou vyloučeny ani paušální platby za některou ze služeb. Stejně je možno přistupovat i k přihlášení se k aplikaci. Když si uživatel přeje být zákazníkem aplikace, zakladatel aplikace může tuto registraci zpoplatňovat.

## 7.1 Založení aplikace bank

V rámci zakládání aplikací může zakladatel taktéž založit banku. Jedná se o obdobný způsob jako ostatní aplikace, avšak bankovní služba je též úzce spjata s automatickou službou centrální banky a aplikací transakce, případně zpoplatněnou službou nebo nějakým typem poplatku. Jedná se opět o tabulku databáze, kde je jako záznam v databázi definován stav konta uživatele banky.

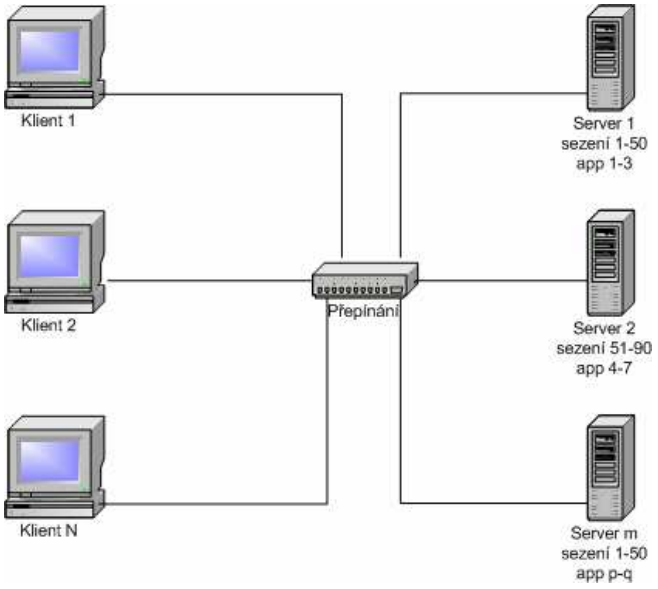
# 8 Existující podobné alternativy

Pokud hledáme alternativy pro tuto práci, nenajdeme v podstatě obdobný projekt. Můžeme však srovnávat dílčí části. Pro toto srovnání pracovně rozdělme tento projekt na více okruhů a ke každému z nich můžeme rovnat alternativy již existujících projektů.

Prvním hlediskem může být část „Úzký klient“. Jde o přístup k programům serveru z terminálu. Na terminálu není instalováno žádné programové vybavení, vše je uloženo na serveru a taktéž zaopatřováno na straně serveru. Jako známějším přístupem k řešení a pravým opakem tohoto přístupu je řešení pod méně známým názvem „Tlustý klient“. [18]

Dalším hlediskem je část „informační systém“. Program situuje veškeré programové proměnné do databáze. Současně dává možnost tuto databázi dále doplňovat existujícími aplikacemi, k nimž mají přístup i specifičtí uživatelé.

Svým přístupem za pomoci terminálu projekt konkuruje i aplikacím umisťovaným na webových stránkách, jako jsou například Java Applety a podobně.




**Obr.25: Řešení sítě klientů [18]**

Řešení hromadné obsluhy klientů na současně více serverech bývá řešeno systémy přepínání, jak je znázorněno na obr. 23., kde přepínač může znázorňovat složitější síťovou strukturu nebo řízenou topologii. [18]

# 9 Možnosti rozšíření

Projekt má spoustu možných oblastí rozšíření, uveďme některé příklady možného navázání na práci. Především potom součásti, které napomohou k praktické použitelnosti projektu. Jedná se o práci, která má široké uplatnění, vzhledem k jejím silným stránkám z hlediska univerzálnosti a základu v kvalitním programovém vybavení.

## 9.1 Grafická nástavba

Vzhledem k tomu, že se jedná o programové prostředí především pro běžné uživatele, očekává se, že aplikace v okně terminálu bude přehledně graficky vyobrazována. Proto jako vhodnou možností rozšíření je grafická nástavba taková, aby uživatel mohl pohodlně procházet menu programu a zadávat příkazy pouze výběrem kurzorovými klávesami. Nástavba může vycházet z klasického terminálového rozlišení 25 krát 80 znaků, případně dynamicky reagovat na změnu velikosti, pokud půjde o terminálové okno.

## 9.2 Ovladače podsystémů

Ovladače jsou nezbytnou součástí spojení systému s hardwarovými součástmi, neboli již dříve zmíněnými řídícími podsystémy. Především v našem případě pro přístup k datovým bodům řídících podsystémů a komunikace mezi podsystémy, je vhodné vytvořit ovladače zařízení, neboli drivery pro tato externí zařízení.

Vhodné pro naši potřebu je rozlišit ovladač zařízení zvlášť pro každý řídící podsystém a současně další ovladač pro každý jeden datový bod řídícího podsystému. Na tomto základě potom můžeme striktně rozlišovat přístup uživatelů pro konkrétní jednoznačně opravňované zařízení.

## 9.3 Další ukázkové aplikace

Programové vybavení je například velmi vhodné pro provozování síťových počítačových her. Některé z dalších ukázkových aplikací je možno směřovat v tomto duchu a využít tak síťového prostředí v kombinaci s navázanou databází a správou klientů.

Další možností aplikace může být uživatelův diář využívající databáze, tedy přenositelná sada tabulek pro zápisky do kalendářové struktury.

Patřičným rozšířením s přidáním množství tabulek je možno postavit na programu informační systém. Podobných řešení se často využívá u některých komerčních produktů, především v nasazeních, kde je vyžadována vysoká spolehlivost.

# 10 Závěr

Práce se věnovala návrhu a realizaci programu, jenž zprostředkovává provoz aplikací uživatelů. Tito uživatelé se typicky přihlašují přes obecné terminály. Je zde rozlišeno několik typů uživatelů, a to podle jejich možných uplatnění. Jejich práva jsou řešena příslušností do uživatelských skupin.

Součástí realizace je i několik ukázkových aplikací, na nichž se dají demonstrovat vlastnosti řešení. Účelem programu je především možnost rozšíření ze strany uživatelů s příslušnými právy. Doplňování aplikací nebo i uživatelů do prostředí není tedy podmíněno zásahem administrátora.

Ukázkové programy prezentují jak komunikaci mezi uživateli, tak možnost přístupu uživatele k zařízení. Vychází se zde z možností nastavení patřičných práv různým uživatelům ze strany vlastníků aplikací, případně také zpoplatnění služeb nebo úkonů uživatelů.

Jsou zde maximálně využity možnosti skupin uživatelů síťových operačních systémů. Pro často proměnlivé položky, jako je například text komunikace a podobně je využito ukládání do databáze.

# Literatura

[1] BRANDEJS, M.: Linux - praktický průvodce. Konvoj, Brno 2003.

[2] GRAHAM, S.; SHAH, S.: Administrace systému Linux - podrobný průvodce začínajícího administrátora. Grada, Praha 2007.

[3] STEVENS, R.W.: Unix Network Programming. Englewood Clifs, Prentice Hall 1990.

[4] BOURNE, A.S.: The Unix System. Addison-Wesley, London 1982.

[5] MADNICK, S.E.; DONOVAN, J.: Operating Systems. McGraw Hill, New york 1974.

[6] BACH, M.J.: Principy operačního systému Unix. Softwarové Aplikace a Systémy, Praha 1993

[7] ČEŠKA M.: Petriho sítě, Akademické nakladatelství Cerm, 1994.

[8] VOZÁR, V.: Model automatizace budov [online], 2005. Dostupný z URL: http://dce.felk.cvut.cz/dolezilkova/diplomky/2005/dp_2005_vozar_vaclav.pdf.

[9] HODNÝ, J.: Dálkové řízení technologického procesu [online], 2004.Dostupný z URL: http://dce.felk.cvut.cz/dolezilkova/diplomky/2004/dp_2004_hodny_jiri.pdf.

[10] WAGO Technical documentation, ELECTRONICC [online]. Dostupný z URL: http://www.wago.com/wagoweb/documentation/index_e.htm.

[11] MARKL J.: Petriho sítě I [online], 2006. Dostupný z URL: http://www.cs.vsb.cz/markl/pn/index.html.

[12] LonMark functional profiles [online]. Dostupný z URL: http://www.lonmark.org/products/fprofile.htm.

[13] ŽÁČEK K.: Úvod do skriptování v Bash [online], 2000. Dostupný z URL: http://docs.linux.cz/programming/interpreted/bashdoc-1.4/index.html.

[14] SKOČOVSKÝ, L.: Principy a problémy operačního systému Unix. Science, Veletiny, Brno 2008.

[15] WELLING, L.; THOMPSON L.: PHP a MySQL - rozvoj webových aplikací, SoftPress, Praha 2003.

[16] MULLER, K.: Výpočetní technika a programování. Vydavatelství ČVUT, Praha 1999.

[17] Manuál k operačnímu systému Debian GNU/Linux [online]. Dostupný z URL: http://www.debian.org/doc/manuals/

[18] SHINDER, D.L.: Computer networking essentials. Cisco Press, Illinois 2001.

# Seznam příloh

# Příloha A

Předvedení tabulek databáze:

```
#!/bin/bash
        DBS=`mysql -uroot  -e"show databases"`
        for b in $DBS ;
        do
                mysql -uroot -e"show tables from $b"
        done
```

Založení uživatele:

```
#!/bin/bash
mysql –u uživatel –e "CREATE USER venca IDENTIFIED BY 'heslo'"
```

Vytvoření tabulky databáze:

```
#!/bin/bash
mysql –u uživatel –e "CREATE TABLE nazev_tabulky (Položka_1 Datový_Typ,
Položka_1 Datový_typ Položka_1 Datový_typ)"
```

Záznam do tabulky databáze:

```
#!/bin/bash
mysql –u uživatel –e "INSERT INTO nazev_tabulky (Položka_1, Položka_1,
Položka_1)
VALUES(‘hodnota_položky_1’, ‘hodnota_položky_2’, ‘hodnota_položky_3’)"
```

Dávkové zpracování:

```
#!/bin/bash
mysql –u uživatel –e "mysql -u karel –p heslo < skript.sql"
```

Zobrazení dat z tabulky databáze:

```
#!/bin/bash
mysql –u uživatel –e "SELECT * FROM tabulka ORDER BY Položka_1, Položka_2"
```

# Příloha B

## Obsah CD:

- ***Elektronický text diplomové práce***

  Soubor: DiplomovaPrace.doc

  Formát Word 97-2003 dokument

  Verze programu MS Office 2003

  Soubor: DiplomovaPrace.pdf

  Formát PDF 1.4

  Verze programu PDFCreator 0.9.5

- ***Obraz virtuálního počítače***

  Adresář souborů: Debian-dipl

  Verze programů na obrazu počítače:

  Debian GNU/Linux 2.6.26-1-686

  MySQL Server 5.0.51a-24

  MySQL Client-5.0 5.0.51a-24

  Formát Typical 5

  Platforma i686 (32 bit)

  Verze programu VMware Workstation 5.5.1