VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Virtuální síťová laboratoř

Diplomová práce

2005 Pavel Němec

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 11. května 2005 .....

Děkuji všem, kteří mne při psaní této diplomové práce podpořili, zejména pak vedoucímu diplomové práce Ing. Petru Grygárkvi, Ph.D. za cenné rady a připomínky.

Abstrakt

Tato práce se zabývá návrhem a implementací informačního systému a přístupového serveru, které jako celek umožňují kompletní správu a procvičování úloh v síťové laboratoři přes webové rozhraní, zejména v době večerních a nočních hodin a dnů bez výuky, kdy laboratoř téměř není využívána. Úlohy využívají síťových zařízení učebny, přičemž student nemusí být fyzicky přítomen v učebně, ale může pracovat na úloze odkudkoli z Internetu. Obecně je možno využít aplikaci pro jakoukoli jinou učebnu s rozličnými zařízeními, ovládanými přes asynchronní sériové rozhraní *RS232*.

Klíčová slova: síťová laboratoř, VirtLab, vzdálený terminál, síťová úloha, správa síťových konfigurací

Abstract

The aim of the work is to design and implement information system and access server, which enable remote acces to network laboratory and work there on selected network tasks. Task consists of network equipments and students must not be in the laboratory, but can work remote using Internet.

Keywords: network laboratory, VirtLab, remote terminal, network task

Seznam použitých zkratek a symbolů

TCP	–	Transmission Control Protocol
HTML	–	HyperText Markup Language
XML	–	eXtensible Markup Language
WWW	–	World Wide Web
ERD	–	Entity Relationship Diagram
SŘBD	–	Systém Řízení Báze Dat

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

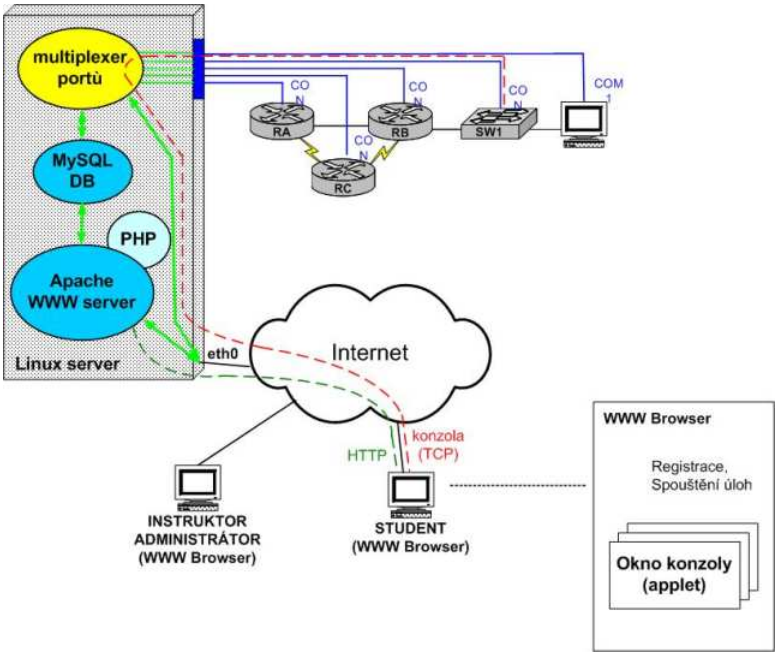
Je známo, že nejlépe se člověk naučí dané problematice praxí. To platí rovněž o zapojování a konfiguraci počítačových sítí, což je náplní výuky mnohých předmětů vyučovaných nejen na vysokých školách. Mnohdy ale studenti nemají dostatek času si vše důkladně prozkoušet při cvičeních v laboratoři, neboť problematika počítačových sítí je velice rozsáhlá. Student může narazit při zapojování a konfiguraci cvičných úloh v době výuky na nejrůznější problémy, jejichž řešením stráví značnou dobu a na probíranou látku tak zůstane jen málo času. Vzhledem k tomu, že vybavení laboratoře bývá poměrně nákladné a ve večerních a nočních hodinách a o víkendech, či jiných dnech bez výuky není laboratoř využita, myšlenka využití učebny i mimo vyučovací hodiny je nasnadě. Realizací této myšlenky je projekt Virtuální síťové laboratoře, řešící vzdálený přístup do laboratoře prostřednictvím sítě Internet.

Princip systému je zachycen na obrázku 1. Ve vrchní části obrázku je ukázka úlohy, jejíž zařízení jsou navzájem propojena. Typickými zařízeními laboratoře jsou směrovače nebo přepínače. Ve většině případů obsahují vlastní operační systém s interpretem příkazů, který představuje komunikační rozhraní. Zařízení jsou připojena k počítači prostřednitvím sériového rozhraní *RS232* (na obrázku znázorněno modrou čarou) a pomocí množiny příkazů je lze konfigurovat. Práce studentů při výuce spočívá ve fyzickém propojení sítě a konfiguraci zařízení právě prostřednictvím počítače.

Kdyby se konzola tohoto počítače dala přenést mimo učebnu, nebo ještě lépe mimo školní budovu, mohl by student přistupovat k zařízení i mimo výuku a odjinud než z laboratoře. A právě vytvoření vzdálené konzoly je jedním z cílů této práce. Přenesení, či prodloužení konzoly je dosaženo softwarovými prostředky.

Projekt Virtuální síťové laboratoře jde však ještě dále a mimo vlastní vzdálenou práci na úlohách umožní sestavit počítačovou síť bez nutnosti přítomnosti fyzické osoby, prostřednictvím elektronického zařízení pro automatické propojování zařízení.[1]

[1] Automatizovaný systém správy síťových konfigurací, diplomová práce obhajovaná v červnu 2005 na katedře 638, FMMI VŠB-TU Ostrava, autor David Seidl.



Obrázek 1: Schéma systému

1.1 Cíle práce

Cílem práce je navrhnout a implementovat systém, který umožní procvičování síťových úloh v laboratoři prostřednictvím Internetu. Dílčími částmi jsou návrh a implementace informačního systému (IS), přístupového serveru, appletu a komunikačního protokolu.

Informační systém zajišťuje správu úloh, rezervací, zařízení, uživatelů a spouštění úloh. Základem IS je databáze a systém dynamických www stránek tvořící komunikační rozhraní se systémem.

Java applet představuje terminál, pomocí něhož je studentům umožněno pracovat vzdáleně se zařízním. Komunikace mezi appletem a serverem je definována komunikačním protokolem, který vedle přenášených dat umožní přenášet i řídící informace.

Přístupový server řídí vlastní přístup k síťovým prvkům. Jedná se o serverový program, ke kterému se applet připojí prostřednictvím *tcp* spojení. Po získání všech potřebných informací vytvoří kanál mezi zařízením a appletem. Studentovi se práce jeví jako by byl přítomen přímo v učebně. Na obrázku 1 je vyznačena komunikace uživatele s IS a kanál pro komunikaci uživatele se zařízením.

Stanice, na níž je možné systém provozovat, je počítač s operačním systémem Unixového typu. Na něm je zprovozněn jak webový server, tak i serverový program pro řízení přístupu k jednotlivým prvkům. Stanice je osazena přídavnou deskou s osmi sériovými porty, skrze než uživatel přistupuje k jednotlivým zařízením. Obecně není počet sériových rozhraní limitován, je možné použít přídavnou desku se 16 porty, nebo vložit do počítače více karet.

1.2 Struktura práce

Při vývoji systému jsou použity prostředky a metody softwarového inženýrství. Řazení jednotlivých kapitol tak odpovídá jednotlivým fázím při procesu vývoje softwaru. V kapitole 2 jsou rozepsány požadavky na systém včetně diagramů případů užití (use case) a definicí rolí uživatelů systému. Kapitola 3 zpřesňuje popis jednotlivých funkcí. Kapitola 4 zpřesňuje funkce systému s přihlédnutím k implementačnímu prostředí. Příloha A pojednává o instalaci systému.

2 Specifikace požadavků

V této kapitole je rozpracován rozbor požadavků na systém, jehož výsledkem je seznam funkcí, které bude systém implementovat.

Na obrázku 2. jsou zachyceny hlavní funkce a role uživatelů v systému v diagramu případu užití.

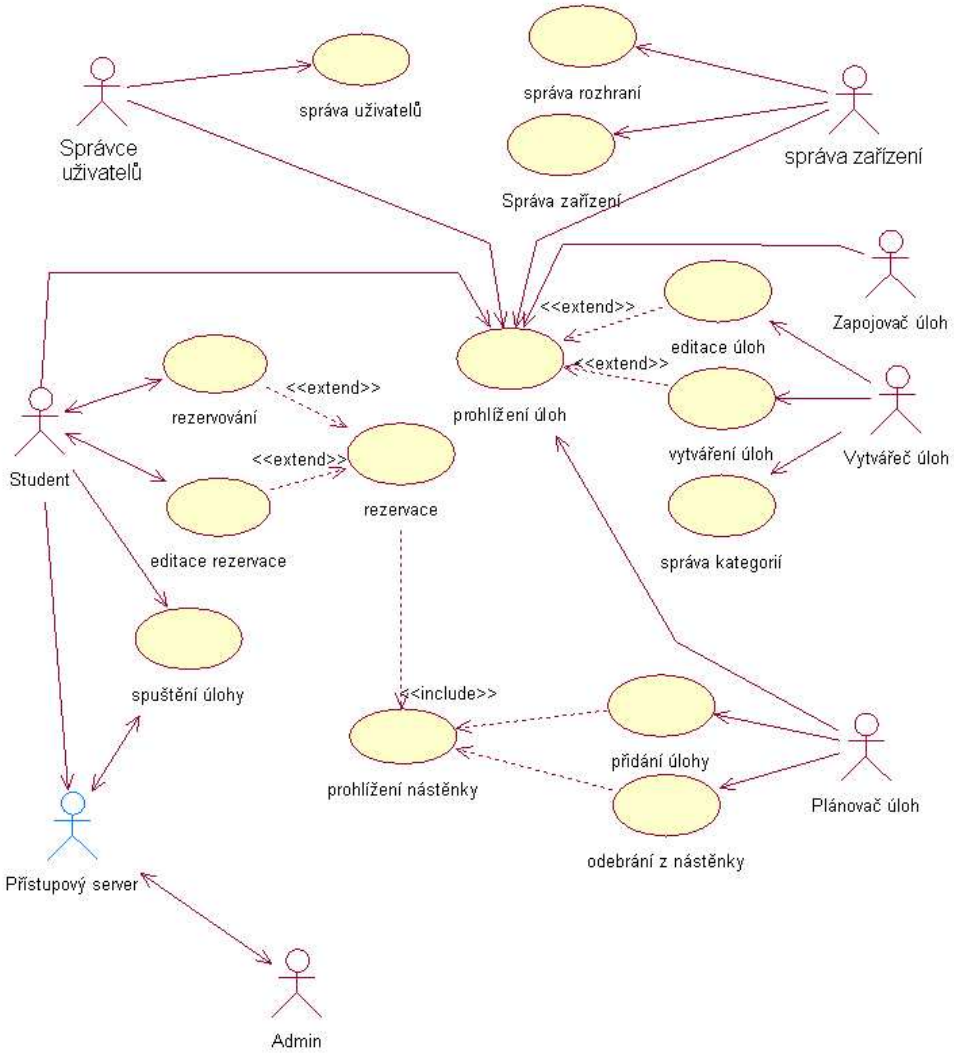
2.1 Úloha

Primárním cílem systému je umožnit studentům vzdáleně konfigurovat zařízení síťové úlohy. Úloha musí být popsána a zanesena do systému, který umožňuje jejich správu, což zahrnuje vytváření, editaci, odstraňování a dále zpřístupňování na elektronicku nástěnku, případně odebírání z nástěnky.

Definice úlohy obsahuje:

- jednoznačný identifikační kód,
- název úlohy,
- seznam zařízení laboratoře použitých v úloze,
- seznam linek, kde každá linka propojuje dvojici zařízení úlohy,
- počtem časových jednotek, který představuje čas, za který by uživatelé měli být schopni nakonfigurovat všechna zařízení a uvést úlohu do provozu,
- obrázek schématu zapojení,
- seznam kategorií, zařazujících úlohy do množiny podobných úloh pro přehlednější vyhledávání,
- popis úlohy,
- ukázkové konfigurace zařízení úloh,
- konfigurační soubor pro automatické propojovací pole,

Úlohy lze do systému vkládat buď vyplněním jednotlivých položek ve webovém formuláři, nebo pomocí zkomprimovaného souboru, který obsahuje především *XML* soubor s definicí jednotlivých údajů, obrázek schématu zapojení a soubor s popisem úlohy (je možno mít popis rozdělen i do více



Obrázek 2: Hlavní diagram případu užití

souborů či adresářů). Vytvářeč úloh (viz. definice rolí uživatelů) si tak může připravit úlohu i bez připojení k Internetu a tu pak vložit do systému.

Popis úlohy obsahuje informace o úloze jako například obecný úvod, odkazy na danou problematiku, návod postupu řešení nebo upozornění na běžné chyby při konfiguraci úlohy. Popis je text formátovaný pomocí *HTML* značek. Může být zadán buď vyplněním položky ve webovém formuláři (zde jsme však omezeni pouze na jeden soubor), nebo vytvořením archivu s více soubory dalších pomocných obrázků, či jiných vložených souborů.

Kategorie

Jak již bylo zmíněno, úlohám mohou být přiřazeny libovolné kategorie, a to i několik kategorií současně. Kategorie jsou řazeny do tříd, které umožňují seskupit podobné kategorie a umožňují přehlednější vyhledávání. Tak např. kategorie *počítačové sítě 1, technologie počítačových sítí, Cisco CCNA 1* mohou patřit do třídy *Vyučovací předměty. Směrování, LAN, WAN* do třídy *Zařazení úlohy.* Při vyhledávání se vyberou úlohy zařazené do požadované kategorie. Pokud je v systému vytvořeno více tříd, je možno vybrat z každé jednu kategorii a vyberou se úlohy patřící do všech vybraných kategorií. Systém umožňuje spravovat kategorie i třídy kategorií(vytvářet, editovat, rušit).

2.2 Zařízení

Systém spravuje informace o zařízeních (síťových prvcích), jež tvoří základní stavební kameny úloh.

O zařizeních se uchovávají tyto informace:

- název(*id*), který je současně jednoznačným identifikátorem,
- sériové číslo,
- obrázek,
- typ, nesoucí informaci, zda se jedná o směrovač, rozbočovač, opakovač, nebo o jiné zařízení
- seznamem rozhraní (sériové porty, ethernetové porty, ...),
- číslo konzoly, specifikuje na který sériový port serveru bude zařízení připojeno,
- poznáka, jež vystihuje současný stav, či může nést jakékoli doplňující informace,

- id užiavatele právě připojeného k zařízení,
- seznamem příkazů, které je nutno vykonat před přihlášením nového uživatele do zařízení. Jedná se například o vymazání stávajících konfigurací, nebo různá nastavení. Těmito příkazy se provede inicializace zařízení.

Aby mohlo být zařízení vzdáleně přistupné, musí být připojeno na fyzické rozhraní serveru (konzolu). Tato informace je důležitá také pro zapojování úloh. Mapování zařízení na konzoly se může měnit a mění se v závislosti na úlohách, neboť v přístupovém serveru bude zřejmě méně sériových portů než je zařízení. Proto musí systém umožnit měnit přiřazení daných portů k zařízením. Je rozumné povolit změnu přiřazení konzol pouze v dostatečném časovém předstihu před zapojením úlohy, aby se nestalo, že některé zařízení úlohy, která má být spuštěna, nemá přidělenu konzolu.

2.3 Rezervační systém a elektronická nástěnka

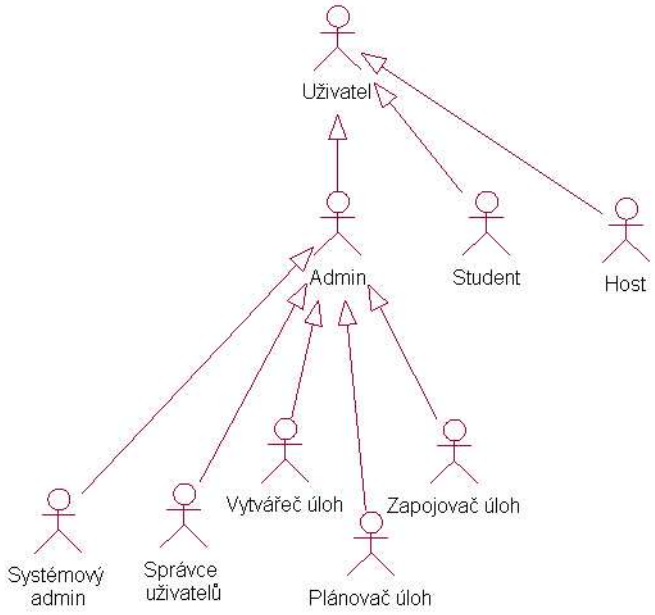
Čas je pro práci s úlohami rozdělen po úsecích, nazvaných *časová jednotka* neboli *timeslot*. Jedna časová jednotka představuje 45 minut, obdobně jako vyučovací hodina na školách.

Rezervační systém umožňuje studentům prohlížet úlohy na elektronické nástěnce a provádět rezervace, popř. rušení rezervací. Na jedné úloze může pracovat více studentů najednou, maximální počet studentů pracujících na úloze je počet zařízení úlohy. Student má při rezervaci možnost určit další lidi, kteří s ním budou spolupracovat, nebo si ponechat celou úlohu pro sebe. Student, který provedl rezervaci, má zvláštní status, jenž jej opravňuje i později přidávat až do povoleného množství další studenty. Avšak nelze již jednou přidané studenty odebírat (i když jim provedl rezervaci). Rezervace mohou být rušeny nejpozději do začátku úlohy. Úlohy z nástěnky lze odebírat, pokud nejsou rezervovány. Existuje výjimka, kdy úlohu je nutno z nějakého důvodu z nástěnky odstranit. Úlohu lze odstranit s tím, že bude uživatel o zrušení jeho rezervace uvědoměn.

Při vkládání úloh na elektronickou nástěnku je vedle vlastní úlohy zaznamenáno, jestli bude úloha v laboratoři zapojena administrátorem, nebo systémem pro automatickou správu konfigurací.

2.4 Role uživatelů

Role uživatelů jsou rozděleny do základních dvou kategorií, administrátoři (správci) a studenti. Ostatním uživatelům je nabídnuta prohlídka systému, tito uživatelé jsou označováni jako *host*. Studenti využívají systému ke studijním účelům. Role uživatelů jsou zachyceny na obrázku 3. U studentů a administrátorů je v systému ukládáno jméno a příjmení, heslo a email.



Obrázek 3: Role uživatelů systému

2.4.1 Administrátor

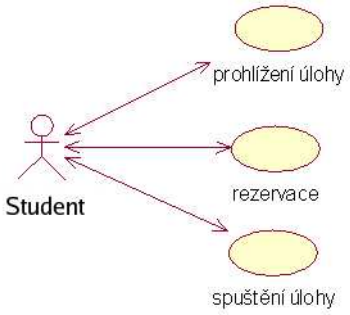
Administrátor má definovánu množinu práv s následujícími vlastnostmi:

- Správa úloh
  zahrnuje vytváření, editace a mazání úloh. Dále má na starosti správu kategorií úloh.

- Správa zařízení
  zahrnuje vytváření, editace a odstraňování zařízení. Dále má na starosti správu seznamu rozhraní, což je tabulka definující možné typy rozhraní (viz. dále).
- Správa uživatelů
  umožní přiřadit uživatelům jednotlivé role, v případě zapomenutého hesla vygenerovat nové.
- Plánovač úloh (nástěnkář)
  má právo přidávat a odebírat úlohy z elektornické nástěnky.
- Zapojovač úloh
  je administrátor, který má na starosti sestavování úloh.
- Systémový správce
  má mimo všechna uvedená práva na starosti funkčnost celého systému.

2.4.2 Student

Role student je v systému důležitá, neboť systém má sloužit právě studentům. Mohou prohlížet, rezervovat nebo spouštět úlohy (obrázek 4). Student má definovánu kvótu, která by měla pomoci spravedlivému rozdělování času při rezervacích. Kvóta představuje počet časových jednotek, jež může během týdne rezervovat. Při rezervaci se tato hodnota dekrementuje. Při dosažení nulové hodnoty již nejsou další rezervace umožněny. Po zrušení rezervace se naopak kvóta inkrementuje. Na začátku každého týdne (v pondělí 00:00 hodin) je kvóta nastavena na výchozí hodnotu.



Obrázek 4: Funkce uživatele *Student*

2.5 Práce na úloze

V době, kterou si student dříve rezervoval, mu je umožněna práce na vybraném zařízení úlohy (popř. více zařízeních). Provede se připojení do zařízení a může začít vlastní práce. Systém kontroluje čas dokdy může být student připojen k zařízení. Při vypršení času, nebo pokud student není po určitý čas aktivní, je od zařízení automaticky odpojen. Deset a pět minut před odpojením je student upozorněn.

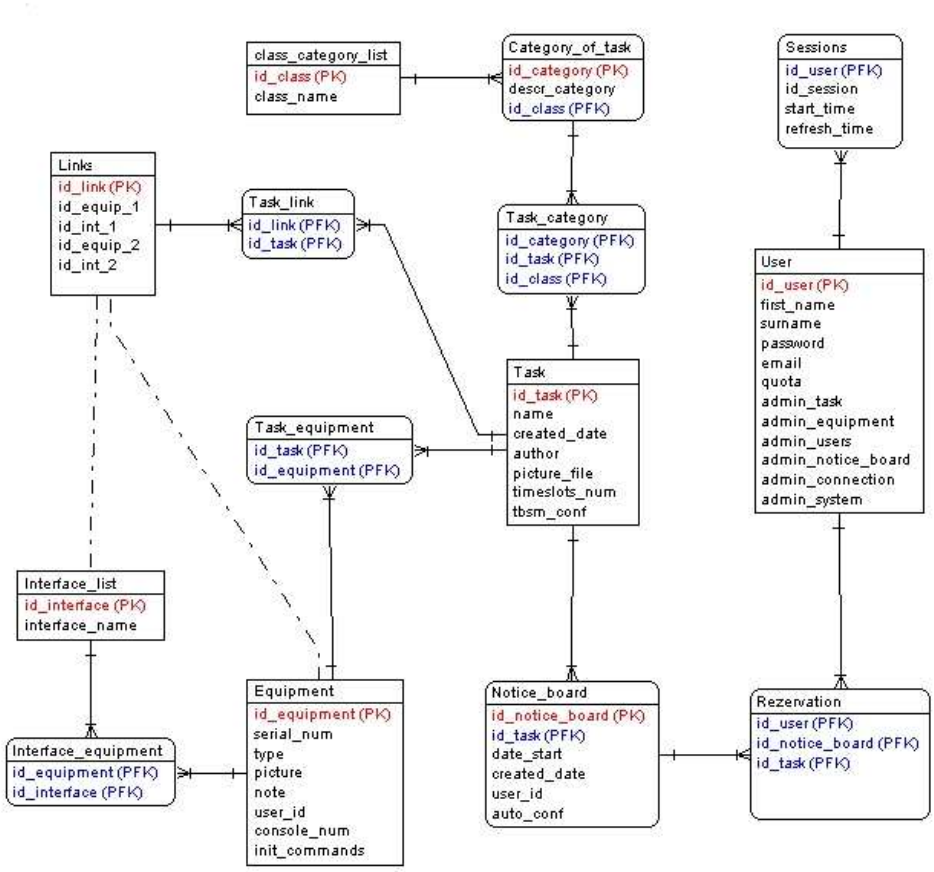
2.6 Další požadavky na systém

- navýšení kvóty na výchozí hodnotu jednou týdně ve stanovenou dobu
- na začátku každého časového úseku zkontrolovat, zda nemá být spušťena jiná úloha nebo zda začne na úloze pracovat jiná skupina studentů. V kladném případě jsou inicializována zařízení. Pokud má být úloha sestavena pomocí automatického propojovacího pole, systém do něj zapíše data z inicializační souboru úlohy.
- Práce se zařízením není nic jiného, než posílání znaků tvořících jednotlivé příkazy. Některé příkazy lze v navrhovaném systému označit za nebezpoečné, neboť mohou znemožnit práci se zařízeními. Nejdůležitějším příkazem je asi nastavení přístupového hesla k zařízení. Běžnou praxí je mít při výuce dohodnutá hesla, která jsou známá. Systém by tedy měl definovat množinu slov nebo slovních spojení, která systém neumožní vložit do zařízení.
- Zjištění, zda je skutečně k dané konzole serveru připojeno očekávané zařízení. Zde bude využita informace o sériovém čísle v definici zařízení.

3 Analýza

V této kapitole jsou upřesněny funkce všech částí systému, definována databáze a princip komunikace mezi appletem a zařízením.

3.1 Databáze



Obrázek 5: ER diagram

Databáze je klíčovým bodem. Jejímu návrhu bylo věnováno značné úsilí zejména při analýze, ale i později byl návrh několikrát doplňován a upravován. ER diagram (viz. obrázek 5) popisuje databázi systému na konceptuální úrovni. Je optimalizován, aby zachovával hlavní databázová paradigmata, jimiž jsou konzistence, neredundance a integrita.

Úlohy (Task)

Úlohy jsou definovány, jak již bylo uvedeno ve specifikaci požadavků, údaji definovanými v následující tabulce.

Field	Type	Null	Key	Default	Extra
id_task	varchar(10)		PRI		
name	varchar(30)	YES		NULL	
created_date	time			00:00:00	
author	varchar(10)				
picture_file	varchar(20)	YES		NULL	
timeslots_num	int(11)	YES		NULL	
tbsm_conf	varchar(20)	YES		NULL	

Id_task je jednoznačný identifikátor úlohy. *Name* je detailnější popis úlohy, *created_date* je čas vytvoření úlohy. *Picture_file* je hlavní obrázek zapojení úlohy a poslední dva atributy představují počet časových jednotek úlohy a konfigurační soubor pro propojovací pole.

Tabulka **Category_of_task** udržuje informace o kategoriích. *Id_category* je jednoznačný identifikátor, *descr_category* představuje název kategorie a *id_class* pak zařazení kategorie do třídy. Tento atribut je cizím klíčem z relace **Class_category_list**, která představuje číselník.

Category_of_task

Field	Type	Null	Key	Default	Extra
id_category	tinyint(4)		PRI	NULL	auto_inc
descr_category	varchar(20)	YES		NULL	
id_class	varchar(20)	YES		NULL	

Class_category_list

Field	Type	Null	Key	Default	Extra
id_class	tinyint(4)			0	
class_name	varchar(30)				

Task_category

Field	Type	Null	Key	Default	Extra
id_category	varchar(20)		PRI		
id_task	varchar(8)		PRI		

Task_equipment

Field	Type	Null	Key	Default	Extra
id_task	varchar(8)		PRI		
id_equipment	varchar(20)		PRI		

Tabulka **Task_category** je vazební tabulka mezi úlohami a kategoriemi. Každá úloha tedy může obsahovat libovolný počet kategorií.

Tabulka **Task_equipment** je vazební tabulkou mezi úlohami a zařízeními. Jak je patrno z ER digramu, úloha může mít různá zařízení a zařízení může být přiřazeno více úlohám.

Zařízení (equipment)

Equipment

Field	Type	Null	Key	Default	Extra
id_equipment	varchar(20)		PRI		
serial_num	varchar(20)		PRI	NULL	
type	varchar(20)	YES		NULL	
picture	varchar(20)	YES		NULL	
note	varchar(512)	YES		NULL	
user_id	varchar(20)	YES		NULL	
console_num	varchar(20)	YES		NULL	
init_command	varchar(512)	YES		NULL	

```
Interface_list;
+----------------+-------------+----+-----+---------+-----+
| Field          | Type        |Null| Key | Default |Extra|
+----------------+-------------+----+-----+---------+-----+
| id_interface   | tinyint(4)  |YES |     | NULL    |     |
| interface_name | varchar(30) |YES |     | NULL    |     |
+----------------+-------------+----+-----+---------+-----+

Equipment_interface
+-------------+------------+------+-----+---------+-------+
| Field       | Type       | Null | Key | Default | Extra |
+-------------+------------+------+-----+---------+-------+
| id_equipment| varchar(8) |      | PRI |         |       |
| id_interface| int(11)    |      | PRI | 0       |       |
+-------------+------------+------+-----+---------+-------+
```

Tabulka **Equipment** obsahuje informace o zařízeních. Zařízení má přidělenu množinu rozhraní, která jsou uložena v číselníku - tabulce **Interface_list**. *User_id* představuje id studenta, který je právě připojen k zařízení, *note* označuje poznámku, *console_num* číslo přidělené konzoly a *init_command* inicializační příkazy pro úvodní nastavení zařízení. Vztah rozhraní a zařízení je modelován vazební tabulkou **Equipment_interface**.

Propojovací linky

```
Link
+------------+-------------+------+-----+---------+----------+
| Field      | Type        | Null | Key | Default | Extra    |
+------------+-------------+------+-----+---------+----------+
| id_link    | int(11)     |      | PRI | NULL    | auto_inc |
| id_equip_1 | varchar(20) |      | PRI |         |          |
| id_int_1   | int(11)     |      | PRI | 0       |          |
| id_equip_2 | varchar(20) |      | PRI |         |          |
| id_int_2   | int(11)     |      | PRI | 0       |          |
+------------+-------------+------+-----+---------+----------+
```

Task_link

Field	Type	Null	Key	Default	Extra
id_task	varchar(10)		PRI		
id_link	int(11)		PRI	0	

Tabulka **Task_link** definuje propojovací linky úlohy. Linka je jednoduše řečeno propojení dvou zařízení pomocí sériového, ethernetového nebo jiného typu kabelu. Musíme mít určena zařízení jež jsou propojena a jejich příslušná rozhraní. Linka je definována identifikátorem *id_link*.

Linka může být obsažena ve víc úlohách, proto je definována vazební tabulka *Task_link* která tyto vztahy zachycuje.

Nástěnka a rezervace

Notice_board

Field	Type	Null	Key	Default	Extra
id_notice_board	varchar(20)		PRI		
id_task	varchar(8)		PRI		
date_start	time			00:00:00	
created_date	varchar(20)	YES		NULL	
user_id	varchar(20)	YES		NULL	
auto_conf	enum(N,Y)	YES		NULL	

Rezervation

Field	Type	Null	Key	Default	Extra
id_notice_board	varchar(20)		PRI		
id_user	varchar(20)		PRI		
id_user_res	varchar(8)		PRI		

Notice_board (nástěnka) uchovává informace o úlohách vložených na nástěnku. Zachycuje čas začátku úlohy, id administrátora, jenž úlohu zpřístupnil a čas vložení do nástěnky. Dále, pro lepší manipulaci s SQL dotazy

obsahuje umělé *id*, jako většina tabulek. A konečně tabulka **Reservation** představuje vlastní rezervační záznamy. Jak plyne ze specifikace požadavků, může pracovat více uživatelů na úloze zároveň, avšak jen uživatel který provedl první rezervaci má status umožňující přidávat dalších spolupracujících uživatelů. *Id_user_res* nese informaci o id uživatele, který provedl zarezervování. *Auto_conf* je zadáváno při vložení úlohy na nástěnku a představuje příznak, zda má být úloha v laboratoři fyzicky sestavena pomocí automatického propojovacího zařízení.

Uživatel

User

Field	Type	Null	Key	Default
id_user	varchar(20)		PRI	
first_name	varchar(20)			NULL
surname	varchar(20)			NULL
faculty	varchar(20)	YES		NULL
email	varchar(20)	YES		NULL
kvota	tinyint(4)			NULL
admin_task	enum('N','Y')	YES		NULL
admin_equipment	enum('N','Y')	YES		NULL
admin_users	enum('N','Y')	YES		NULL
admin_notice_board	enum('N','Y')	YES		NULL
admin_connection	enum('N','Y')	YES		NULL
admin_system	enum('N','Y')	YES		NULL

Tabulka **User** uchovává informace o užiavatelích. Jedná se o jméno, příjmení, heslo, a další důležité informace. Heslo je vhodné ukládat jako šifrovaný text.

Sessions

Field	Type	Null	Key	Default	Extra
id_session	char(20)		PRI		
user_id	char(10)		PRI		
start_time	datetime	YES		NULL	
refresh	datetime	YES		NULL	

Tabulka **Sessions** uchovává informace o přístupech uživatelů do systému. Obsahjue vlastní hodnotu této proměnné, id uživatele, čas vytvoření záznamu a čas posledního přístupu do systému.

Pomocné programy

Pro implementaci požadavků z druhé kapitoly je třeba definovat další pomocné programy, které zajišťují:

- navyšování kvóty na začátku každého týdne,
- na začátku každé časové jednotky zkontrolovat zda má začít nová úloha. Máli být spuštěna nová úloha, program zjistí z databáze, jestli mají být zařizení propojena automaticky. Pokud ano, načte z databáze konfigurační soubor propojovacího pole a pošle data do propojovacího zařízení.
- pokud budou stávající úlohu konfigurovat jiní studenti, do zařízení se pošlou konfigurační informace

3.2 Funkce systému

3.2.1 Přihlášení do systému

Při přihlášení uživatele je provedena jeho autentizace pomocí jména, hesla a je vygenerován jedinečný kód pro sezení, který se používá pro další ověřování jeho totožnosti při požadavcích na další stránky. Jedná se o obdobu běžných *sessions*. V tabulce *Sessions* jsou uloženy jednotlivé kódy spolu s id uživatele. Je třeba ošetřit dobu sezení uživatele, aby se nestalo, že by byla zbytečně zaplňována tabulka Sessions v databázi, pokud se uživatel neodhlásil. Kód sezení se uloží do databáze. Zároveň s ním je do tabulky *Sessions* uložen aktuální čas přihlášení do položek *start_time* a *refresh*. Vždy, když systém autentizuje uživatele při přístupu do kterékoli stránky, je provedena kontrola jeho kódu sezení a v kladném případě je aktualizována položka *refresh* v tabulce *Sessions* nastavením na aktuální čas. Při přihlášení uživatele a generování nového kódu sezení proběhne kontrola všech kódů sezení a jejich doba platnosti. Všechny, ke kterým nebylo dlouho přistupováno, se odstraní. Kontrolou doby platnosti se rozumí rozdíl *start-time* a *refresh* položek v tabulce. Je-li rozdíl větší než definovaná doba pro neaktivitu uživatele, je položka vymazána. Takto je ošetřena situace, kdy se uživatel regulérně neodhlásí ze systému.

Výsek záznamu z tabulky Sessions je uveden níže.

Ukázka záznamu v tabulce Sessions

```
+-----------+---------+--------------------+-------------------+
| id_session| user_id | start_time         |refresh            |
+-----------+---------+--------------------+-------------------+
| 143879798 | nem114  | 2005-03-24 10:59:04|2005-03-24 10:10:04|
+-------------+------------+---------------+-------------------+
```

3.2.2 Úlohy

Vytvoření úlohy

Úlohy lze do systému vkládat dvěma způsoby. Pomocí webového formuláře nebo jako XML soubor. V prvním případě je třeba vyplnit všechny položky webového formuláře. Popis úlohy lze vepsat buď do textového okna, včetně formátovacích značek HTML, nebo načíst soubor s popisem. Při načtení popisu jako *upload* souboru, může být soubor s popisem běžný textový soubor (s příponou .txt, nebo .html), nebo zkomprimovaný soubor (s příponami .tgz nebo .zip). Výhodou komprimovaného souboru je možnost přiložit do popisu více souborů s texty, obrázky, případně mít popis rozdělen do více adresářů. Zkomprimovaný soubor je po načtení rozbalen do adresáře s úlohou. Pro zobrazení vložených údajů v prohlížení úloh je nutno mít jeden hlavní soubor

popisu, který je pojmenován **hlavni.html**. Název hlavního souboru je jednou z proměnných definovaných jednoznačně v celém systému.

Úloze lze přiřadit libovolé množství kategorií. V praxi se bude zřejmě jednat o několik málo kategorií obvykle ne více než pět. Struktura databáze však umožňuje přidat více kategorií. Po kliknutí na tlačítko přidat kategorii ve webovém formuláři je přidána nabídka seznamu kategorií, ze kterého si uživatel vybere.

Dalším prvkem v definici úlohy jsou linky (ekvivalentní název propojení). Je třeba definovat, jak jsou zařízení úlohy propojena. K tomu nám slouží tabulky *Link* a *Task_link*. Linky jsou pro zjednodušení zadávány do textové oblasti formuláře jako jednotlivé řádky s definovanou syntaxí. Ovšem systém k tomu pomáhá nabídkami zařízení a rozhraní. Linka je zadána následujícím řetězcem.

```
zarizeni_1:rozhrani_1, zarizeni_2:rozhrani_2;
```

Příkladem může být *RA:serial2, RB:serial0/1;*. V tabulce *Link* jsou tyto údaje uloženy spolu s id linky, a vazební tabulka *Task_link* uchovává informace o přiřazení jednotlivých linek úlohám. Při vytváření úlohy je prohledána tabulka linků, zda již není potřebná linka zanesena v databázi. Pokud ano, id nalezené linky je uloženo do databáze (do vazební tabulky *Task_link*) spolu s id úlohy. Jinak je nejdříve vytvořena nová linka, která se použije.

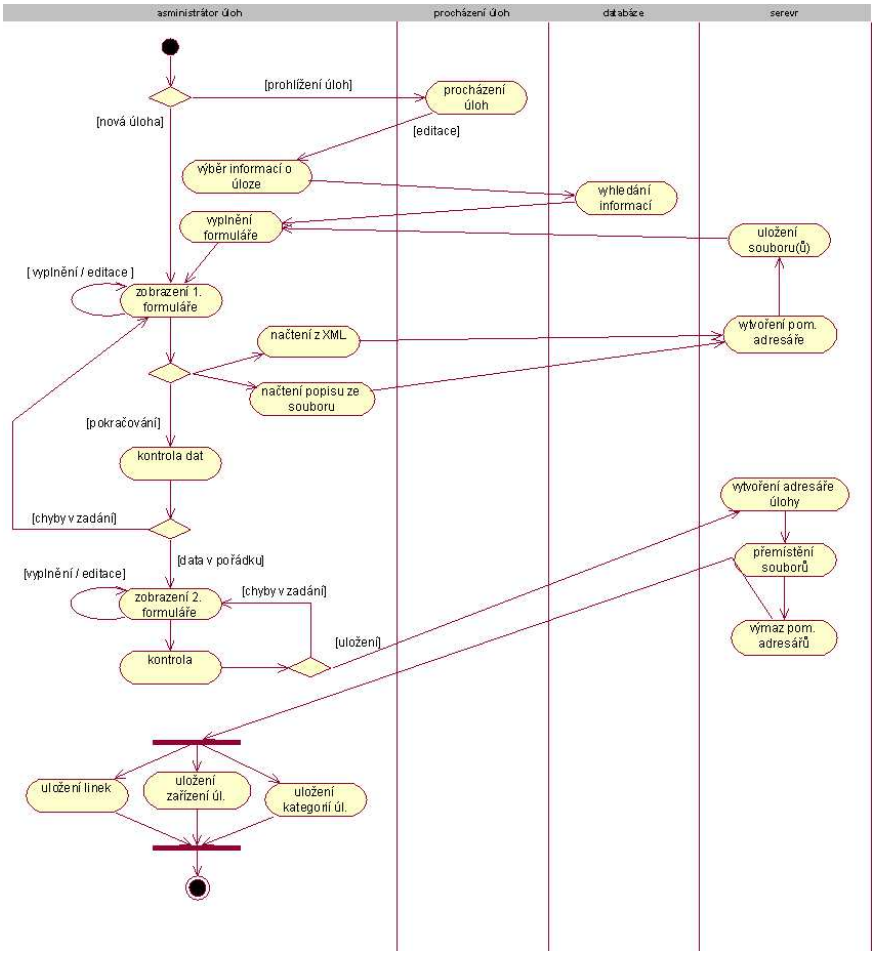
Vložení úlohy pomocí XML souboru

Pro vložení je třeba mít vytvořený soubor s XML popisem. Struktura souboru je popsána v DTD *uloha.dtd* (viz. příloha B) a je ekvivalentní vkládání úlohy pomocí webového formuláře. XML soubor spolu s obrázkem úlohy, konfiguračním souborem pro propojovací pole a popisem úlohy (buď jednoduchý, nebo formou adresářů, případně komprimovaný s více soubory) lze vložit pouze jako archiv s příponou .tgz, nebo .zip.

Archiv úlohy při vložení pomocí XML obsahuje:

- hlavní xml soubor s údaji (main.xml),
- soubor s obrázkem,
- soubor s konfigurací pro propojovací pole,
- popis úlohy (jediný soubor, nebo archiv).

Editace úloh je práce totožná s vytvářením. Před editací je však formulář pro novou úlohu vyplněn údaji a změněn nadpis stránky. Jinak se postupuje jako při vytváření úlohy.



Obrázek 6: Diagram aktivit zachycující editaci a vytvoření úlohy

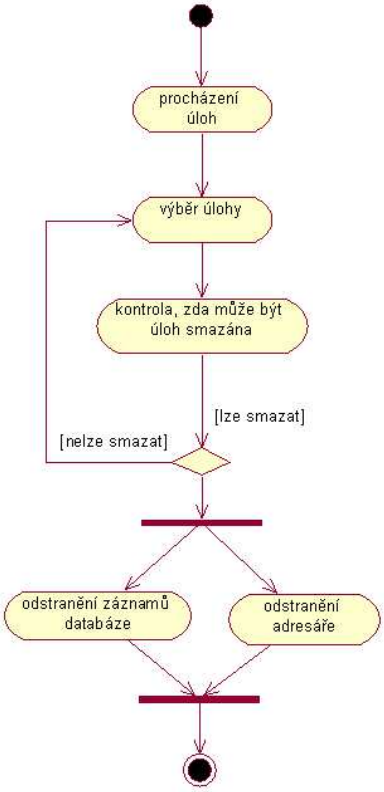
Vytváření a editaci úloh zachycuje diagram aktivit z obrázku 6. Vertikální rozdělení je tvořeno jednotlivými prvky zodpovědnými za provedení operací. Na začátku je proveden výběr úloh vyhledáním v databázi a vyplnění příslušných položek ve formuláři, jak je to popsáno v předchozím odstavci, nebo zobrazení prázdného formuláře pro novou úlohu. Dále lze vložit úlohu pomocí XML souboru, nebo popis pomocí archivu, provede se načtení daných souborů do pomocného adresáře a načtení XML dat nebo popisu do formuláře. Při přechodu na další formulář, kde se zadávají vzorové konfigurace zařízení, je zkontrolováno, zda jsou zadána správná data. Přesněji, zda položky *id* a *name*, nejsou prázdné a zda počet čas. jednotek obsahuje pouze číslice. Dále lze pokračovat už vlastním uložením úlohy tak, že je vytvořen adresář s id úlohy, jsou do něj přesunuty všechny soubory z pomocného adresáře úlohy a nakonec jsou do databáze zaneseny informace o úloze, jejích zařízeních a linkách. Při zpracování XML dokumentu je použita knihovna PHP pro podporu XML. Zde je nutno vytvořit funkce, které jsou volánym když parser narazí na začátek nebo konec elementu, atributy a znaková data. Tyto funkce jsou registrovány u parseru.

Odstranění úlohy

Při odstraňování úlohy je potřeba provést tyto kroky:

1. Zjištění, zda není úloha dána na nástěnku, s časem větším nebo rovným než je aktuální čas. Pokud je tato podmínka splněna, pokračuje se bodem 2.
2. odstranění příslušných záznamu z tabulky *Task*,
3. odstranění potřebných záznamů z tabulky *Task_equipment*,
4. odstranění adresáře pojmenovaného podle id úlohy,
5. odstranění příslušných záznamů z vazební tabulky kategorií,
6. odstranění příslušných záznamů z vazební tabulky linek.

Odstranění úlohy je přehledně zobrazeno na obrázku 7.



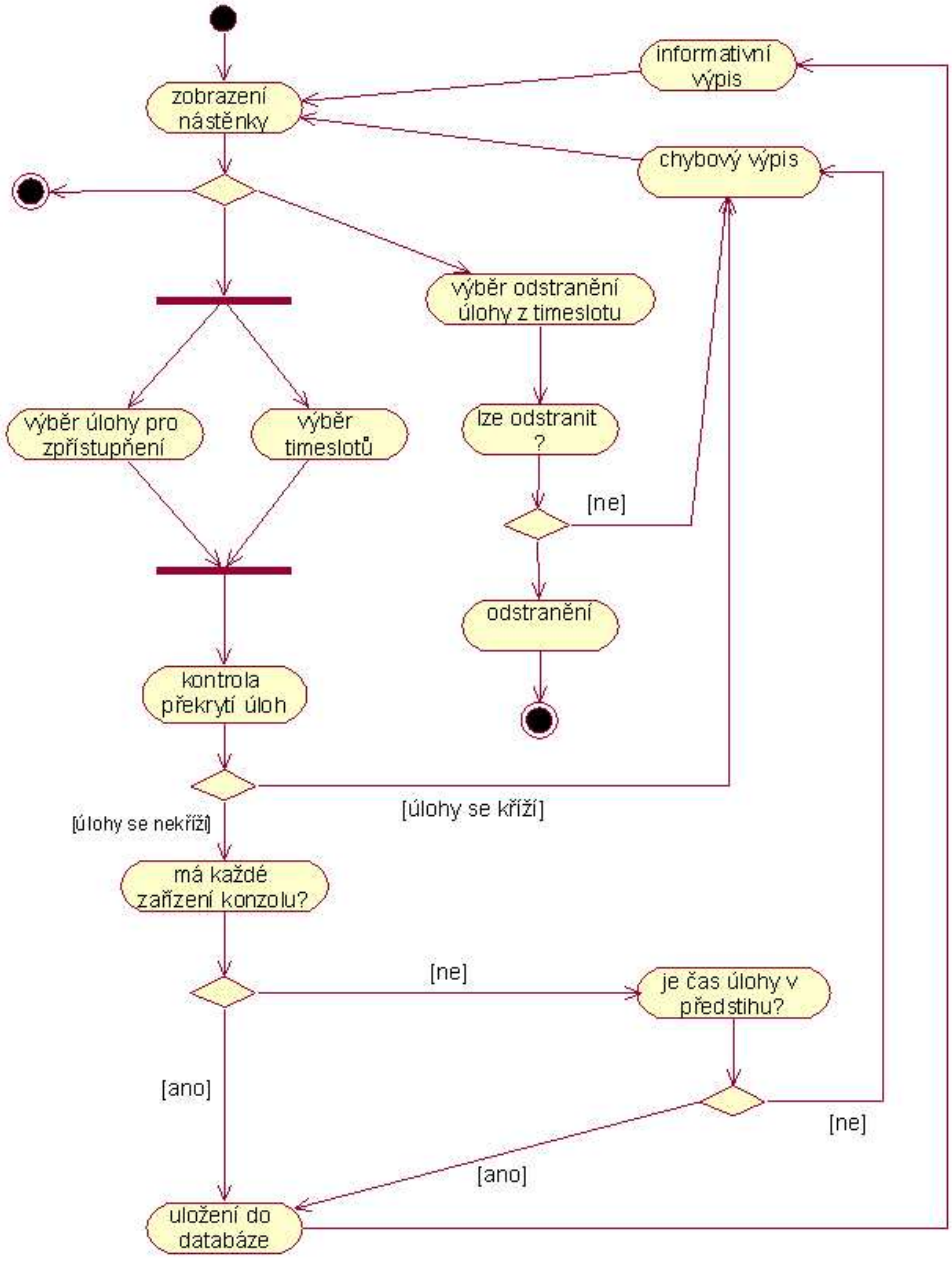
Obrázek 7: Odstranění úlohy

Vložení úlohy na nástěnku.
Je-li timeslot prázdný (není v něm úloha), je možno provést přidání úlohy na nástěnku. Chceme-li přidat další úlohu do již obsazeného timeslotu, je to možné pouze v případě, že se nepřekrývají konfigurace úloh. Konfigurace úlohy je množina zařízení úlohy a seznamu linek úlohy. Každé zařízení má přiděleno číslo sériového portu (číslo konzoly ke kterému je připojeno). Podmínkou pro přidání úlohy je, aby každé zařízení úlohy mělo přiděleno číslo konzoly. Pokud nemá, je vypsáno varovné hlášení, že zařízení nemá přidělenu konzolu a dále se postupuje v závislosti na timeslotu, kam má být úloha vložena. Je-li timeslot časov vzdálen o více než 24 hodin dopředu od aktuálního času, je úloha přidána do nástěnky s tím, že je opět vypsáno varovné hlášení oznamující, že je úloha vložena takto podmínečně i bez přiděleného čísla konzoly. Tím je umožněno, aby uživatel mohl vkládat na nástěnku úlohy s časovým předstihem i to několik dní, nebo týdnů bez nutnosti zjišťovat, které zařízení má přidělenu konzolu. Je ovšem nutné minimálně hodinu před začátkem úlohy každému zařízení číslo konzoly přidělit. Při přihlášení administrátora je zkontrolováno, zda úloha, která má být spuštěna do 24 hodin nevyžaduje přiřazení konzoly. Pokud ano, je vypsáno varovné hlášení. Na přidání úlohy do neprázdného timeslotu se vztahují následující podmínky, které jsou pro názornost formulovány matematicky.

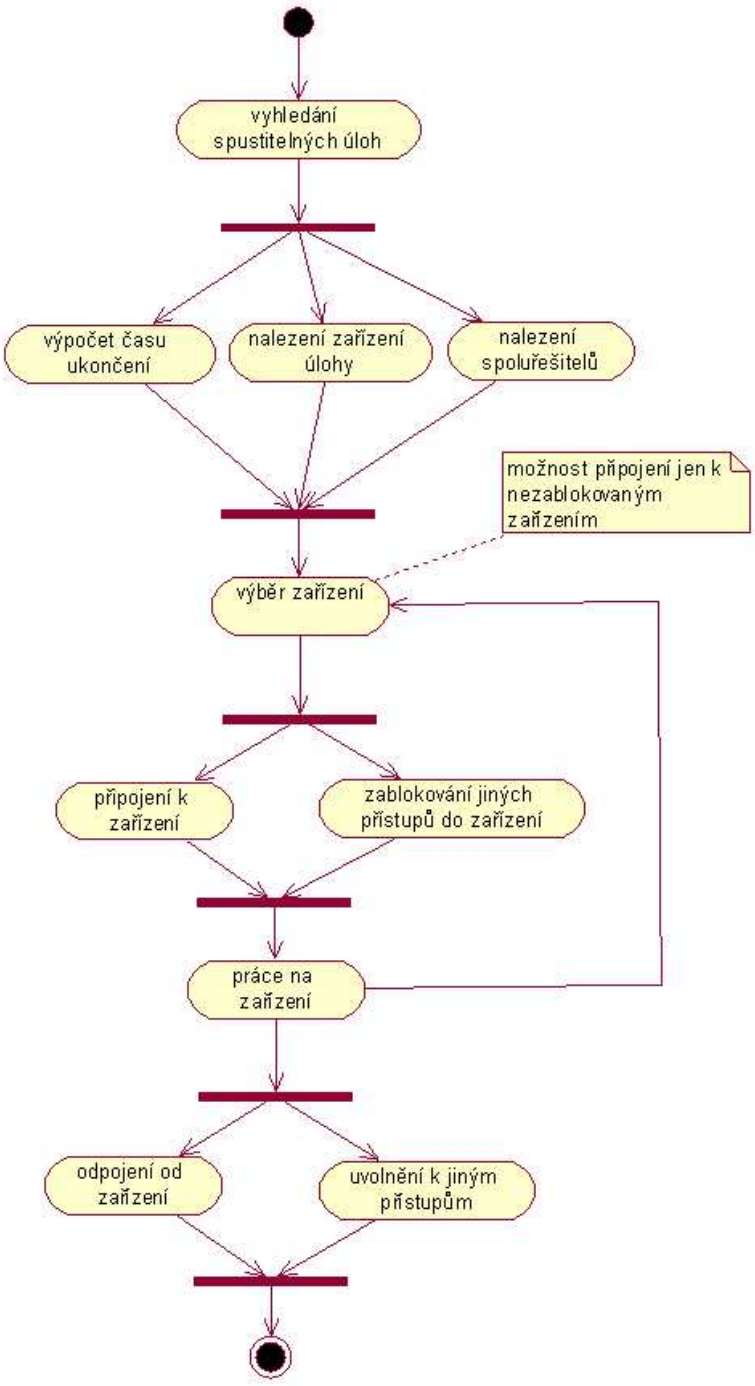
Úkoly, které lze provádět nad nástěnkou zachycuje obrázek 8. Po vstupu na nástěnku lze vybrat odstranění úlohy, nebo vložení úlohy. Pak je provedena kontrola zda lze úlohu do nástěnky přidat a v kladném případě lze provést uložení.

Spuštění úlohy
Po poklikání na odkaz v hlavní nabídce jsou v databázi vyhledány úlohy, které lze právě spustit (pravděpodobně se bude většinou jednat o jednu až dvě úlohy). Je spočten souvislý časový úsek, představuje limit práce na zařízení. Studentovi je nabídnut seznam zařízení úlohy. Pro každé zařízení je zjištěno, zda není zablokováno a zda má přidělenu konzolu serveru. Zablokováno je tehdy, pokud má zařízení přiděleného uživatele, který je právě připojen. Číslo konzoly musí být zařízení přiřazeno, aby server mohl vytvořit komunikační kanál. Je-li zařízení zablokováno, je tlačítko se zařízením ve stavu "disabled" a vypsáno id uživatele, který je k zařízení připojen.
Je-li číslo konzoly rovno **-1**, je tlačítko rovněž ve stavu "disabled" a vypsáno vyrovné hlášení, že zařízení není připojeno na konzolu.

Po výběru je uživatel připojen k zařízení. Nyní může začít vlastní práce. Spuštění úlohy přehledně zachycuje schéma na obrázku 9.



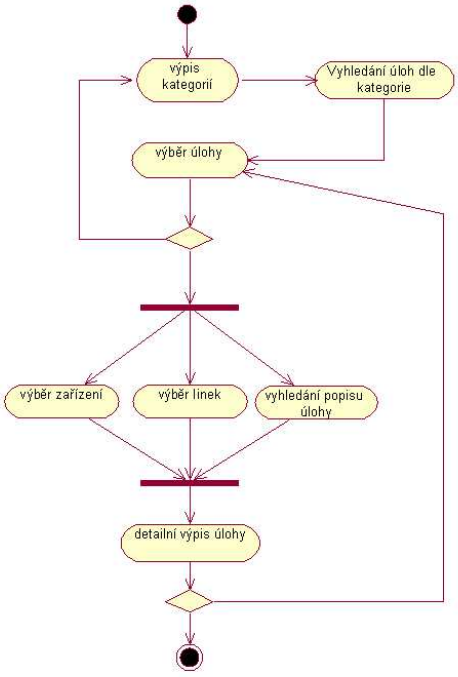
Obrázek 8: Elektronická nástěnka, vkládání a odebírání úloh



Obrázek 9: Diagram činností - spuštění úlohy

Kategorie a prohlížení úloh

Při prohlížení a vyhledávání úloh nabízí systém funkci pro výběr na základě kategorií. Situaci zachycuje diagram aktivit na obrázku 10. Nejdříve je uživateli nabídnut seznam kategorií zařazených podle jednotlivých tříd. Uživatel si může vybrat z každé třídy jednu nebo všechny kategorie. Následně jsou vyhledány úlohy, které jsou přiřazeny vybraným kategoriím. Uživatel si nyní vybere úlohu poklepáním na odkaz s úlohou a poté jsou vypsány všechny detaily úlohy. Administrátoři mají navíc možnost úlohu editovat nebo odstranit.

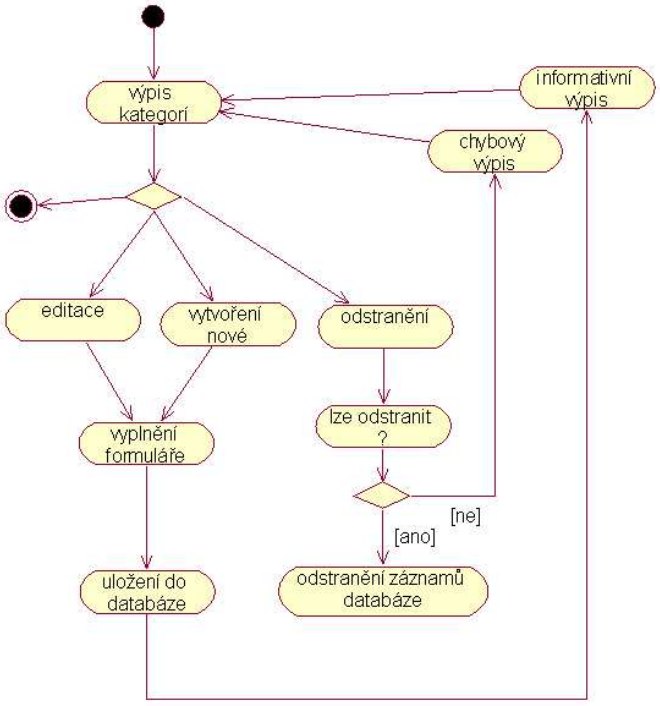


Obrázek 10: Diagram aktivit vyhledávání a prohlížení úloh

Kategorie úloh lze odebírat nebo modifikovat. Při odstranění úlohy je třeba zkontrolovat, zda není přidělena k úloze. Pokud ano, není odstranění povoleno. Protože systém kategorií pracuje nad jedinou tabulkou, obsahuje tabulka kategorií atribut třída (class). Při vyhledávání úloh se postupuje následovně. Pro každou třídu kategorií vyhledáme množinu jejích kategorií *descr_kat*, ta je nabídnuta uživateli k výběru. Uživatel si z každé třídy vybere z listového seznamu jednu kategorii. V případě že z některé třídy nevybere

nic, berou se v úvahu všecny kategorie třídy. Dotaz na úlohy je pak prováděn tak, že se vyhledají úlohy patřící do všech vybraných kategorií.

Má-li uživatel práva pro vytváření úloh, jsou mu u každé nalezené úlohy nabídnuta tlačítka pro editaci a odstranění úlohy. Výjimku představuje úloha, která je v danou chvíli dána na nástěnku. V takovém případě jsou obě tlačítka pro zachování konzistence ve stavu "disabled".



Obrázek 11: Diagram činností - kategorie

3.2.3 Rezervace

Při přístupu do rezervací je studentovi zobrazena elektronická nástěnka. Nástěnka je v podstatě přehledný kalendář, který zobrazuje údaje po týdnech. V řádcích nástěnky jsou dny (Pondělí – Neděle) a ve sloupcích jsou časové úseky počítány v násobcích jednotky času úlohy (45 minut).

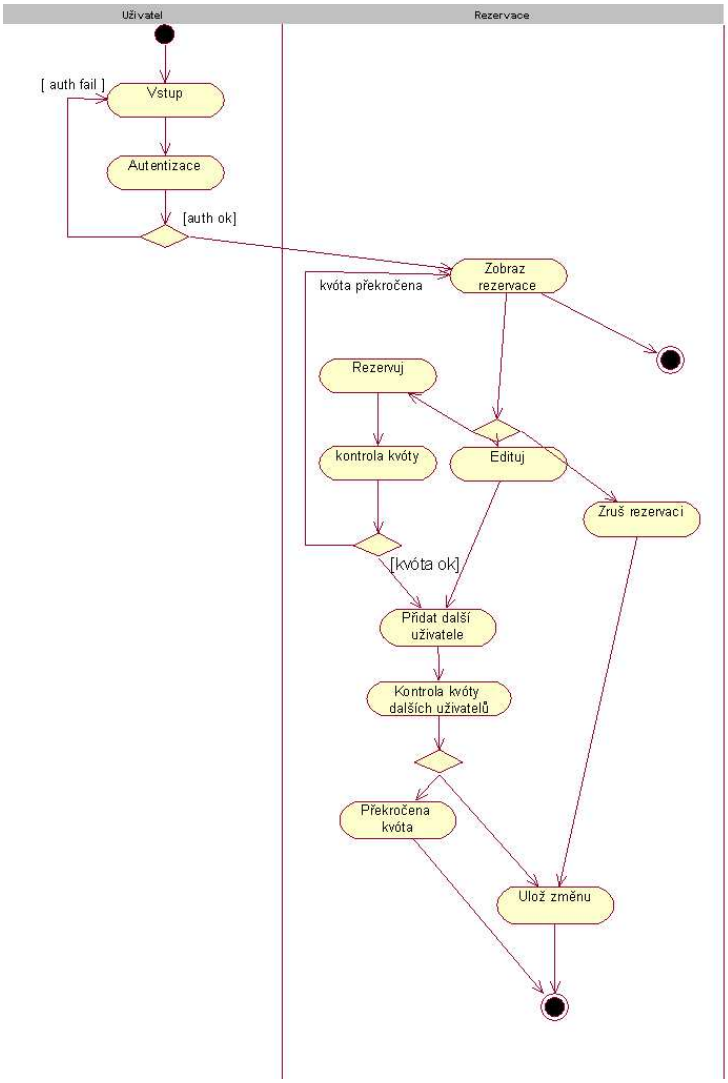
Nástěnka obsahuje několik typů údajů:

- *prázdné políčko*
  není přidělena žádná úloha,
- *rezervace*
  hypertextový odkaz oznamující, že úlohu je možno rezervovat poklepáním na odkaz,
- *id uživatele* jako prostý text
  úloha je rezerována uživatelem s daným id,
- *id uživatele* tučně zvýrazněn modrou barvou
  úlohu rezervoval uživatel s daným id,
- červeně zvýrazněn hypertext. odkaz s textem*delete*
  uživatel, jenž prohlíží nástěnku je uživatelem s tímto id uživatele. Po kliknutí na odkaz lze rezervaci zrušit,
- modře zvýrazněn hypert. odkaz s klíčový slovem *edit*
  uživatel je vlastníkem rezervace, je mu umožněno přidávat další uživatele.

Rezervaci lze provést kliknutím na odkaz *rezervace*. Uživateli je nabídnuto rezervovat další uživatele. Maximální počet uživatelů je počet zařízení úlohy. Pokud neurčí nikoho dalšího, má úlohu jen pro sebe. Dále je vyhledán v databázi nejdelší souvislý časový úsek, jejž lze pro úlohu nastavit, což zjednodušuje práci uživatele. Je-li např. úloha dána na nástěnku pro 8 hodin (následujících po sobě) a standardní doba úlohy je 3 hodiny, spočte se pro uživatele (při poklepání na první hodinu rezervace) úsek 8 hodin, ten je v kombinaci s délkou úlohy upraven na 3 a nabídnuto ukončení po 1., 2., nebo třetí hodině. Hodinu ukončení lze vybrat pro každého přiděleného uživatele.

Podmínkou uložení rezervace je kvóta s hodnotou větší než nula. Při uložení rezervace je kvóta dekrementována pro každou rezervovanou hodinu pro každého uživatele. Při rezervaci o více než jeden týden dopředu a následnému zrušení rezervace v dalším týdnu se kvóta navýší nejvýše na maximální povolený týdenní limit. Tím znemožněno přenesení kvóty do dalšího týdne.

Rezervaci přehledně znázorňuje diagram činností (Obrázek 12).



Obrázek 12: Diagram činností — rezervace

3.2.4 Zařízení

Každé zařízení má definováno číslo konzoly, což představuje sériové rozhraní serveru, na kterém je zařízení připojeno. Je zřejmé již ze specifikace požadavků, že zařízení je více než sériových portů. Proto bude třeba konzoly přidělovat podle potřeby právě těm zařízením, které budou v nejbližší době spuštěny. Nejbližší dobou se myslí čas do 24 hodin před spuštěním úlohy.

Při vytváření nebo editaci zařízení je možno přiřadit číslo konzoly. Není-li zařízení přiřazena konzola, má v položce konzola v databázi hodnotu -1. Chcme-li přiřadit skutečné číslo konzoly, je to možné provést při vytváření nebo editaci zařízení. Jsou vyhledána všechna volná čísla konzol a nabídnuta k přidělení. V přípdě, že by byly všechny konzoly obsazeny, je třeba jednu konzoli uvolnit přidělením hodnoty -1 a uvolněnou pozici pak využít pro další účely.

Při výpisu seznamu zařízení se vedle vlastních odkazů se zařízeními vypisuje číslo konzoly, ke kterému je zařízení právě připojeno.

Při editaci zařízení je zjištěno, zda zařízení není právě využíváno právě běžící úlohou. Bere se časový interval v rozmezí jedné hodiny před začátkem spuštění úlohy a konce úlohy. Takto se vyhledají zařízení úlohy která budou spuštěna nejpozději za jednu hodinu. Těmto zařízením není možné provést změnu konzoly.

Při odstranění zařízení je konrolováno, zda není zařízení přiřazeno některé úloze. V kladném případě odstranění není povoleno.

Vytvoření zařízení

Zařízení lze přidat pomocí webového formuláře. Lze nadefinovat mimo již dříve definované položky seznam rozhraní zařízení. Dále vybrat z nabídky typ, do kterého zařízení patří. Mezi základní typy patří *router, switch, hub, others.*

3.2.5 Uživatel

Nad tabulkou User lze provádět vkládání, vyhledávání, popř. rušení a přiřazování jednotlivých práv. Operace jsou povoleny jen administrátorovi s právy pro administraci uživatelů.

3.3 Přístupový server

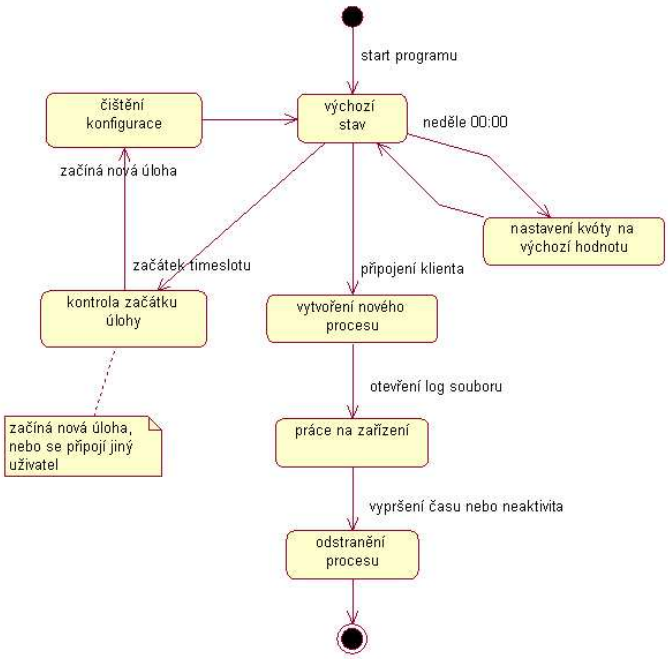
Obrázek 13: Přístupový server

Hlavními funkcemi serveru je

- vytvoření socketu pro připojování klientů,
- připojení klienta,
- autentizace klienta,
- kontrola, zda má klient právo přístupu k zařízení,
- otevření speciálního souboru sériového zařízení,
- přenos dat od zařízení ke klientovi a obráceně,
- kontrola času přístupu uživatele a příslušné reakce na vypršení času,
- zpětné uvolnění zařízení po odpojení klienta,
- odpojení klienta,
- všechny aktivity monitorovat a ukládata do log souborů.

Všechny aktivity serveru je vhodné monitorovat. Za tímto účelem jsou specifikovány dva soubory. Jeden zaznamenává přihlašování uživatelů, navazování komunikací a další aktivity serveru, a další soubor pak vlastní přenášená data. Formát pro každý záznam je uvozen kombinací jména uživatele, data a času události a nakonec vlastní data.

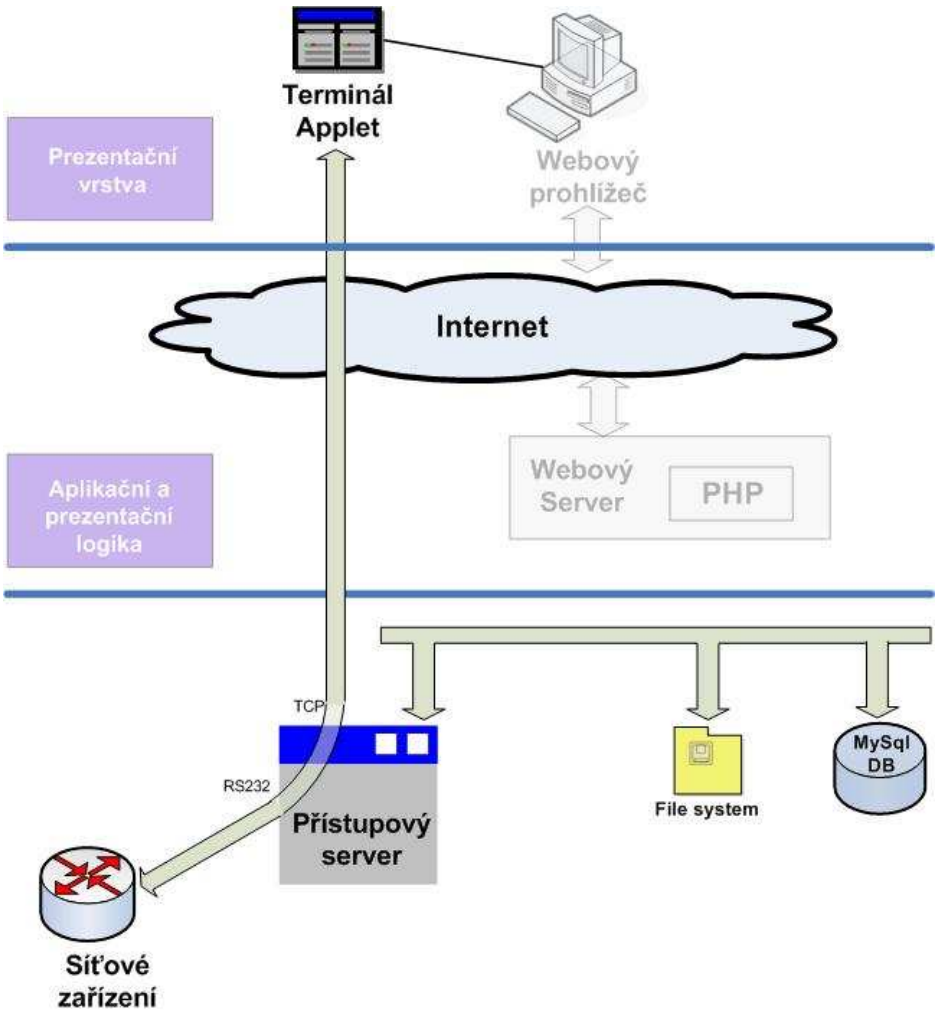
Pčístupový server spolu s pomocnými programy, který výše popsané funkce obstarává, prochází řadou stavů, které jsou zachyceny na obrázku 14.



Obrázek 14: Stavový diagram pro přístupový server s pomocnými programy

3.4 Komunikační protokol

Jak již bylo několikrát řečeno, primárním cílem je vytvoření vzdáleného terminálu, který přes komunikační kanál propojí uživatele se zařízením. Celou situaci výstižně popisuje obrázek 15.



Obrázek 15: Komunikace appletu se zařízením

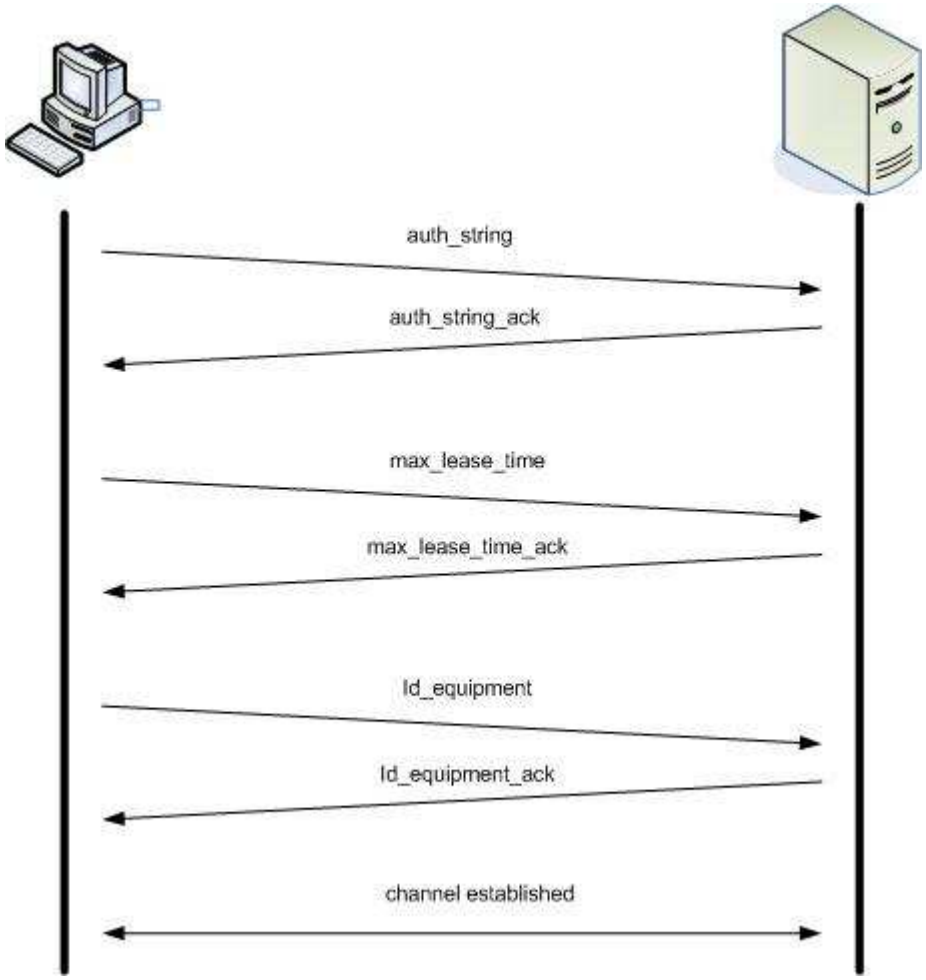
Komunikace je řízena pomocí protokolu, jenž umožňuje přenášet vedle dat i řídící informace. První byte přenášených dat tvoří hlavičku nesoucí tyto informace. Byte hlavičky je rozdělen na dvě části. Nejvyšší čtyři bity představují základní dělení, další čtyři bity pak konkrétní akci. Například pro navazování komunikace jsou nejvyšší 4 bity rovny 0xF, a přenášené informace jsou kódovány do nižších 4 bitů čísly 0 – 15 (dekadicky). Kódy 0–63 (3Fhex)

jsou řídící informace, 64-127 (40-FF hex) tvoří ostatní data. Mezi další řídící informace patří odpojení uživatele při nekativitě, nebo oznámení že za 5 a 10 minut bude student odpojen na základě dosažení časového limitu práce na zařízení. Student tak má dostatek časového prostoru pro uložení konfigurace, případně jinou činnost před odpojením.

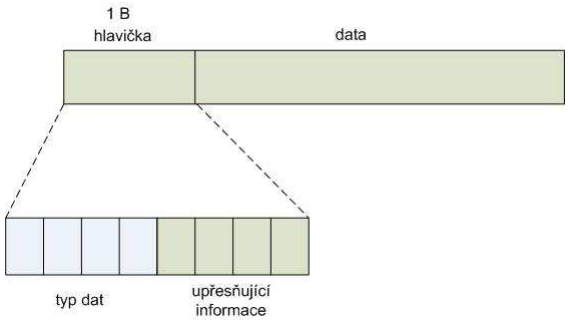
Formát přenášených dat zachycuje obrázek 17. Jednotlivé řídící kódy přenosu jsou zachyceny v příloze D

Princip navázání komunikace je následující (viz. obrázek 16):

1. navázání komunikace se serverem pomocí TCP spojení,
2. klient pošle serveru svůj autentizační řetězec, který mu byl vygenerován www serverem,
3. server ověří v databázi jeho platnost, v kladném případě pošle zpět kladnou odezvu,
4. klient pošle serveru *id* zařízení, ke kterému se chce připojit,
5. server zkontroluje v databázi zda není k zařízení připojen někdo jiný, pokud ne, uloží do databáze *id* uživatele a pošle odpověď klientovi,
6. klient dále posílá čas, dokdy může být připojen k zařízení a sériový port serveru, k němuž je zařízení připojeno. Mezi každým bodem posílá zpět potvrzení o správnosti přijetí,
7. server ověří čas, dokdy může být uživatel připojen a otevře znakové zařízení sériového portu.



Obrázek 16: Navázání komunikace



Obrázek 17: Formát přenášených dat

4 Návrh implementace

Tato část práce definuje rozdělení systému na moduly, dále obsahuje podrobné definice a upřesnění všech funkcí systému.

Implementačním prostředím je operační systém unixového typu. Přístupový server využívá volání služeb operčního systému a je využito pomocných programů pro srpávnou funkčnost celého systému.

4.1 Moduly systému

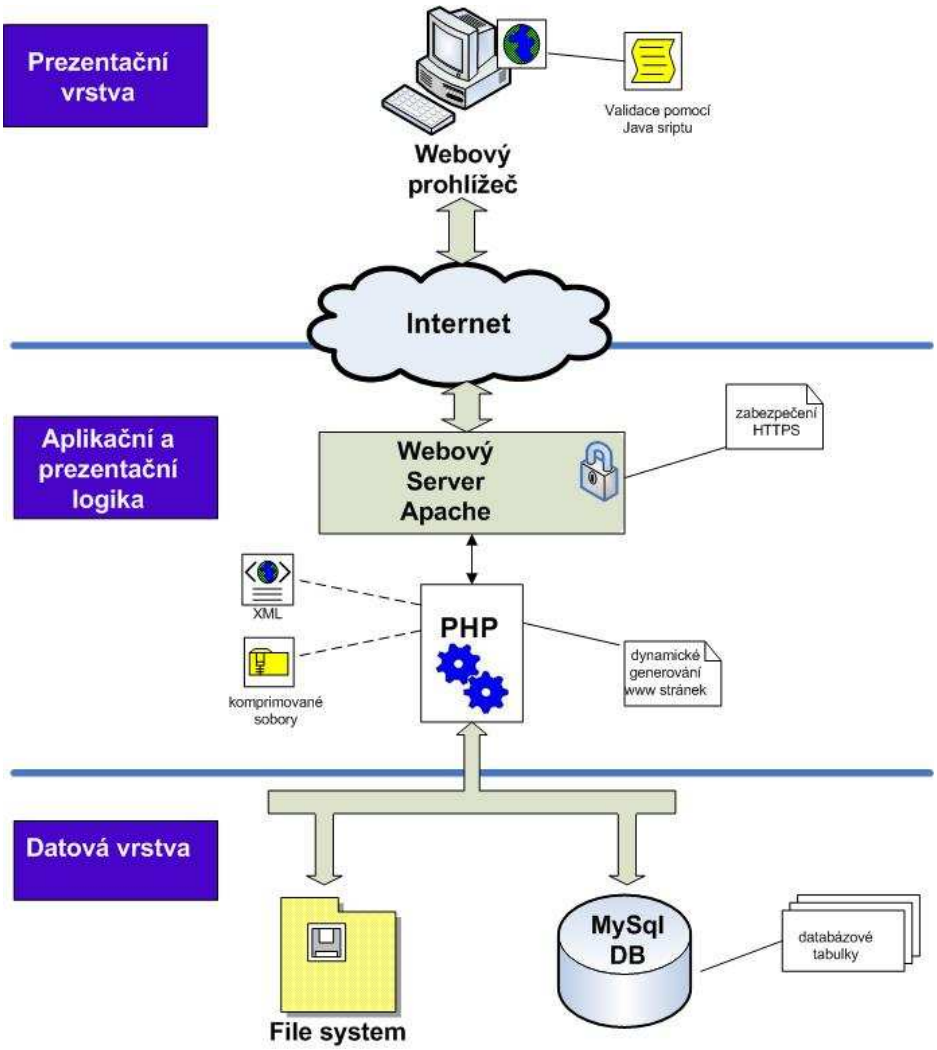
Systém lze rozdělit do několika modulů, které jsou napsány pomocí odlišných technologií a programovacích jazyků. Při výběru technologií byl brán zřetel na požadavek použít volně šiřitelné programy, které jsou šířeny pod hlavičkou GNU, tedy *open source* programy.Rozdělení systému na moduly umožní implementovat a testovat každou část nezávisle.

- Informační systém (IS)
  Jako webový server byl vybrán Apache, o jehož nesporných výhodách není nutné se příliš rozepisovat. Snad stojí za zmínku to, že velká část Internetu běží právě na Apache. Použitý databázový systém je *MySql*. Logika webu je vystavěna na jazyce *PHP*, který je spolu s databází *MySql* provozován úspěšně na velkém množství serverů po celém světě.

- Přístupový server
  Přístupový server (dále jen server) je část systému, která zajišťuje propojení Java appletu v prohlížeči uživatele a vlastního zařízení (prvku úlohy). Server je naprogramován v jazyce *C*. Systém je vytvořen pro operační systém typu Unix.

- Vzdálený terminál - applet
  Prvek, který se stará na klientské straně o komunikaci se serverem. Pro jeho implementaci byl vybrán programovací jazyk Java, který umožňuje jednduché začlenění appletů do webových stránek.

Systém je rozdělen do několika vrstev, jak je běžné při tvorbě inforačních systémů. Každá vrstva spolupracuje se sousední vrstvou pomocí definovaného rozhraní. To umožňuje zakrýt implementační podrobnosti a použité technologie. Tento přístup umožní popřípadě měnit jednotlivé komponenty systému bez nutnosti zasahovat do ostatních částí systému. Situaci zachycuje obrázek 18.



Obrázek 18: Rozdělení systému do spolupracujících vrstev

Nejnižší, datovou vrstvu tvoří databáze a souborový systém. Tato vrstva je zodpovědná za fyzické uchovávání informací a zpracovává požadavky na data, které přicházejí z prostřední vrstvy. Jak již bylo zmíněno, databázový stroj (SŘBD) byl vybrán *MySql*.

Prostřední vrstva představuje aplikační logiku celého systému a prezentační logiku. Jádrem je webový server *Apache* s podporou zpracovávání skriptů *PHP* a *SSL*. PHP stroj využívá knihoven pro zpracování XML souborů, šifrování dat a archivování dat. Prezentační logika využívá kaskádových stylů *CSS*.

Poslední, třetí vrstvu, tvoří tenký klient, kterým je webový prohlížeč s podporou Javy.

4.2 Přístupový server

Implementačním jazykem pro přístupový server je jazyk *C*. Program využívá vedle běžných knihoven funkce knihovny *libmysql* pro přístup do databáze *MySql*.

V OS typu Unix jsou *ttyS* znaková zařízení pro sériové terminálové linky. Komunikace mezi zařízením a serverem přes sériový port je zajišťována pomocí *file descriptoru* sériového portu, který je vrácen voláním funkce *open*. Čtení a zápis je prováděn pomocí běžných funkcí pro čtení a zápis.

Při startu serveru je vytvořen *tcp socket*, na kterém server očekává příchozí požadavky. Při navázání spojení novým klientem je vytvořen nový socket a následně nový proces voláním funkce *fork*, který se stará o klienta po celý zbytek sezení. Původní socket tak může obsloužit další příchozí požadavky.

Po vytvoření procesu jsou *file descriptory* (fd) sériového portu a socketu vloženy do struktury *fd_set*, která je předána funkci *select* jako parametr. Tato funkce kontroluje, zda není na některém *file desciptoru* aktivita. Pokud některý *fd* vykazuje aktivitu čtení nebo zápisu, přečtou se data z jednoho descriptoru (ať ze socketu či sériového portu) a zapíšou se do druhého. Funkce *select* má nastaveno, že při neaktivitě 1 minutu (*timeout*) je proveden blok následující za voláním *select* jako v případě aktivity. Zjistí se ale, že se jedná o *timeout* a je dekrementována hodnota času pro neaktivitu. Po 15 minutách neaktivity (hodnota kontroly posledního přístupu je nula) je klient odpojen. Při každé aktivitě je vždy doba neaktivity navýšena na výchozí hodnotu.

4.3 Java applet

Student komunikuje s přístupovým serverem prostřednictvím Java appletu. Vytvoří socket a pokusí se navázat komunikaci se serverem podle výše naznačeného postupu. Applet je vlastně terminál posílající každý znak stisknutý na klávesnici uživatele na druhou stranu tcp spojení a zároveň na tomto socketu čte příchozí data, která vkládá do textového okna. Čtení ze socketu zajišťuje samostatné vlákno, jež se stará o zachycení události oznamující, že jsou připravena nová data ke čtení. Po přečtení nových dat jsou tato vložena na konec terminálového okna.

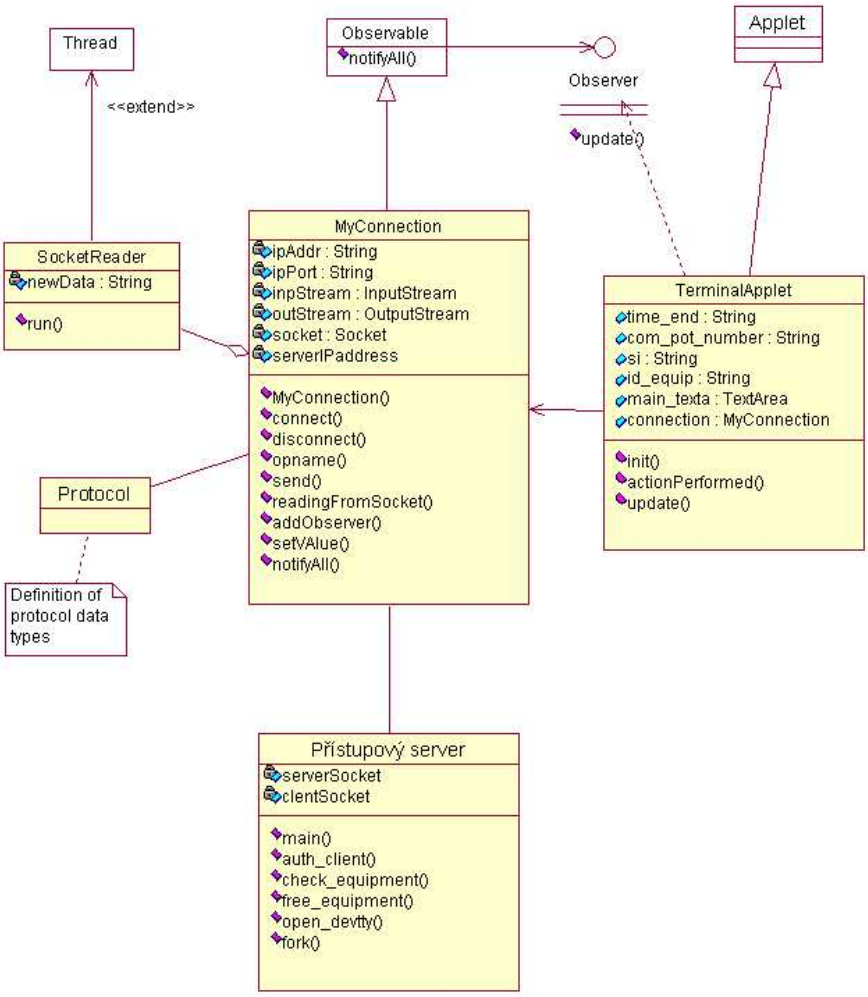
Při spuštění úlohy studentem je vytvořena nová stránka s appletem. Všechny potřebné parametry jsou do stránky předány jako běžné parametry stránky, které si applet načte ze stránky pomocí funkce *getParameter*. jedná se o tyto parametry:

- ip adresa a port přístupvéo serveru,
- číslo konzoly,
- id zařízení,
- čas dokdy může být uživatel připojen v systému,
- session uživatele pro ověření totožnosti a přístupu.

Applet se skládá z několika spolupracujících objektů. Vycházím z objektového návrhu zobrazeném na obrázku 19. Hlavní třída **TerminalApplet** je potomkem třídy *Applet*. Může tak být vložena do webové stránky. Pro každý předaný parametr obsahuje objekt jeden atribut. Při startu appletu je zavolána metoda *init()*, která provede následující kroky:

1. Vytvoří grafické uživatelské rozhraní se všemi prvky. Mezi nejdůležitější patří tlačítka pro připojení a odpojení a textové pole představující terminálové okno.
2. textovému oknu přiřadí naslouchadlo zachycující události stisknutí kláves.
3. Vytvoří objekt třídy *MyConnection*, který se stará o komunikaci s přístupovým serverem a zavolá jeho metodu pro připojení k serveru.

Terminál je simulován textovým oknem, objektem třídy **TextArea** grafické knihovny AWT, které má přidáno naslouchadlo stisku kláves *KeyAdapter*. Toto textové okno má nastavenu hodnotu *setEnabled* na *FALSE*, čímž



Obrázek 19: Třídní diagram appletu a serveru

mu znemožním přímý zápis uživatelem. To je nahrazeno právě naslouchadlem a veškeré aktivity nad oknem má ve své režii třída appletu. To taky znamená, že nebude zobrazován kurzor, proto je kurzor nahrazen znakem "|", který má v režii též applet.

KeyAdapter je abstraktní třída, která implementuje rozhraní *KeyListener*. Ovšem neimplementuje těla metod, popisuje pouze jejich hlavičky. Rozhraní *KeyListener* má definovány tři metody, kterými jsou *KeyTyped*, generující událost při stisku klávesy následovaném puštěním klávesy. *KeyPressed* je voláno při stisku klávesy a *KeyReleased* generuje událost při puštění klávesy. Těla těchto metod jsou implementovány právě až ve vloženém objektu adaptéru naslouchadla.

Běžný terminál umožní pracovat pouze s naposledy vloženým řádkem. Z toho plyne celá řada vlastností, které je třeba ošetřit. Applet při zachycení stisku klávesy zkontroluje o jakou klávesu se jedná. Jedná-li se o běžnou neřídící klávesu, je vytvořen aplikační datový paket, který má v hlavičce příznak že se jedná o běžná data, a poslán objektu třídy *MyConnection* který se postará o přenos k zařízení. Jedná-li se o řídící klávesy, je situace o něco komplikovanější. Je třeba ošetřit klávesy *TAB, ENTER, PAGE UP, PAGE DOWN, kurzorové šipky, CTRL-C, CTRL-D* a další řídící klávesy. Protože Java pracuje se šestnáctibitovými znaky, některé kódy generované Javou se neshodují s kódy Linuxu. Tato situace je ošetřena přemapováním příchozích a odchozích znaků. Ošetření uvedených kláves je specifické pro každou klávesu.

Řešení je následující:

- ENTER
  Do okna terminálu je vložen znak nového řádku a stejně tak do zařízení na druhém konci kanálu. Provedou odřádkování.

- ←
  V Javě je nejdříve smazán kurzor a následně je vložen o jednu pozici vlevo. Odstranění je provedeno přepsáním prázdným řetězcem. Do terminálu je poslán znak BACKSPACE. tedy znak s kódem 8.

- →
  Zde je situace o něco složitější. V Javě vložím do aktuální pozice znak posunu vpravo, znak s kódem 37.

- BACKSPACE
  Vymazání znaku je provedeno tak, že smažu znak před aktuální pozicí a vše od aktuální pozice do konce řádku posunu o jeden znak vlevo. Technicky je to provedeno uschováním znaků od aktuální pozice do

konce řádku, přemazáním všeho do konce řádku a návrat zpět plus jeden znak navíc a zpětné zapsání dříve uschovaných znaků. V Javě jednoduše přemažu znak před kurzorem prázdným znakem.

- DELETE
  Zde je provedena obdoba předchozího příkazu s rozdílem, že se nejedná o znak před kurzorem, ale za kurzorem.

- HOME Řešení je totožné se řešením pro klávesu kurzorové šipky vlevo s rozdílem že za aktuální pozici se bere začátek řádku.

- END Řešení je totožné se řešením pro klávesu kurzorové šipky vpravo s rozdílem že za aktuální pozici se bere konec řádku.

Třída **MyConnection** je zodpovědná za komunikaci se serverem. Mezi hlavní funkce patří navázání TCP spojení se serverem, dále navázání komunikace pomocí inicializačních dat předaných třídou appletu, posílání dat serveru a odpojení od serveru.

Zde je s výhodou využit návrhový vzor **Observer**. Třída TerminalApplet implementuje třídu **Observer**, a implementuje zde metodu *update()*. Tato metoda zkontroluje zda se jedná o řídící data, ta zpracuje, nebo textová data, ta zapíše do okna. Třída MyConnection dědí z Třídy Observable. Při jejím vytvoření je zavolána metoda *readingFromSocket*, která vytvoří nové vlákno, to má na starosti číst nově příchozí data ze socketu. Třída terminalApplet je pozorovatelem třídy MyConnection. V případě, kdy jsou k dispozici nová data na socketu, vlákno je přečte a zavolá metodu *notifyAll(newData)* s novými daty, čímž dá vědět všem pozorovatelům že byly na socketu přijata nová data. To znamená vyvolání metody *update()*.

4.3.1 Bezpečnost

Protože se jedná o síťovou aplikaci, je potřeba zajistit bezpečnost. Je použit zabezpečený protokol *HTTPS*. Při přenosu dat přes kanál vytvořený mezi uživatelem a serverem je rovněž nutno provést šifrování dat.

4.3.2 Úlohy

Pro každou úlohu je v systému souborů (*filesystem*) vytvořen adresář pojmenovaný podle id úlohy. Jsou do něj uloženy pomocné soubory, a to obrázek úlohy, popis úlohy (více souborů či adresářů s popisem úlohy), konfigurační soubor pro propojvací pole a soubor s ukázkovými konfiguracemi. Posledně jmenovaný soubor je pojmenován **solution.conf** a obsahuje ukázkové konfigurace všech zařízení úlohy.

5 Závěr

V této práci je provedena analýza a implementace software pro projekt Virtuální síťové laboratoře. V průběhu návrhu byl systém rozdělen do několika komponent, pro každou určeno implementační prostředí. Analýza byla průběžně rozšiřována a upravována. Zejména ER diagram prošel řadou změn při postupném rozšiřování požadavků nejen ze strany zadavatele, ale vznikajících při samotné analýze.

Jedním z požadavků bylo seznámit se s multiportovýi kartami. Po prozkoumání nabízejících se řešení byl vybrán model *Moxa C168H/PCI* s osmi sériovými asynchronními porty *RS232* a konektor *OPT8-RJ45*, který rozděluje výstup konektoru *DB62* karty na 8 portů *RJ-45*, což je zejména vhodné pro využití ve školní laboratoři.

Informační systém a přístupový server lze provozovat na počítači s operačním systémem Unixového typu. Grafické rozhraní IS je optimalizováno pro prohlížeče *Microsoft Internet Explorer* a *Mozilla Firefox*. Systém byl vyvíjen a testován na platformách Debian Linux a Fedora Linux Core 2.

Výsledky práce byly prezentovány na výroční konferenci institucí zapojených do Cisco Networking Academy, která se uskutečnila 7.-9. dubna 2005 v Brně.

Jelikož se jedná o systém určený pro výukové účely, rozhodli jsme se veškeré podklady pro realizaci a použití jeho softwarové i hardwarové části poskytnout pro nekomerční využití volně k dispozici pod licencí obdobnou licenci GNU. Kdokoli tedy může systém nekomerčně používat a rozšiřovat s jediným omezením, že výsledek rozšíření musí rovněž být dán za stejných podmínek k dispozici. O spolupráci již projevilo zájem několik pracovišť, mezi jinými Technická univerzita v Košicích, Ostravská univerzita, Slezská univerzita v Karviné a SPŠ elektrotechnická v Bratislavě. S těmito institucemi zvažujeme i možnosti realizace distribuované virtuální laboratoře, která by jako jeden celek zpřístupňovala zařízení umístěné na uvedených pracovištích.

5.1 Rozšiřování systému

Sučasný stav projektu je zaměřen na samostatné procvičování úloh. Vhodným rozšířením by byla práce pod dohledem vyučujícího, který by vzdálené cvičení řídil.

Možnost vlastního komunikačního prostředku pro spolupracující studenty. Tato možnost již byla zkoumána, podobně jako předchozí bod, již při analýze. Vzhledem k rozsáhlosti projektu nebyla tato možnost dále zpracovávána. Studenti mohou využít několika volně dostupných prostředků pro vzájemnou komunikaci při práci na úlohách.

Pro pokročilé studenty může být přínosné dálkově provádět vlastní, jimi samými navržené experimenty. K tomu účelu je vhodné dovolit těmto studentům, aby sami v rezervovaném čase dálkově rozhodli o topologii rezervovaných prvků, kterou chtějí propojit. Při použití automatizovaného spojovacího pole toto není technickým ani organizačním problémem.

6 Literatura

[1] MATTHEW, N., STONES R. *Linux začínáme programovat* Praha : Computer Press 2000, s. 477-507. ISBN 80-7226-307-2

[2] MATTHEW, N., STONES R. *Linux programujeme profesionálně* Praha : Computer Press 2001. ISBN 80-7226-532-6

[3] WILLIAMS, H., LANE, D. *PHP a MySQL: vytváříme webové databázové aplikace* Praha: Computer Press 2002. ISBN 80-7226-760-4

[4] KUČERA, M., PETERKA, J., a kolektiv. *Programování na webu. Druhé rozšíení a přepracované vydání* Praha : Mobil Media a.s., 2003. ISBN 80-86593-36-3

[5] CASSTAGNETTO J., RAWAT, R., SCHUMANN, S., SCOLLO, Ch., VELIATH, D. *PHP Programujeme profesionálně* Praha : Computer Press, 2002. ISBN: 80-7226-310-2

[6] MIKLE, P. *XCSS-CSS1, CSS2, CSS2.1-úplná přesná referenční příručka* Brno : Zoner software, s.r.o. 2004. ISBN 80-86815-13-7

[7] HAWLITZEK, F. *Java 2, příručka programátora* Praha : Grada Publishing, spol. s.r.o. 2002. ISBN 80-247-9060-2

A Instalace systému

Popis systému předpokládá nainstalován server Apache, interpret jazyka PHP s knihovnami pro XML a MySql a databázi MySql. Dále Javu, a překladač GNU C, případně CC, G++ a utilitu Cron.

Databáze

Vytvoření databáze pomocí skriptů **createDb.sql** a **fill_db.sql**

```
/* přihlášení do databáze mysql*/
mysql -u admin_mysql -p heslo_admin_mysql

/*vytvoření databáze, tabulek a definice uživatele user_vnl s heslem */
/* "heslo_vnl" se všemi právy pro databázi virt_net_laboratory */
mysql> \. db/createDb.sql

/* naplnění tabulek databáze ukázkovými daty */
mysql> \. db/ fill_db . sql
```

Výpis 1: Vytvoření databáze

Java applet

Zdrojové kódy jsou uloženy v adresáři src/applet Překlad se provede příkazy

```
/* překlad zdrojových souborů */
javac *.java

/* vytvoření archivu ap.jar */
jar -c -m popis.txt -f ap.jar *.class

/* Mámeli vygenerován certifikát , digitální podepsání jar archivu */
/* myKeystore je úložiště s certifikátem , myCert je certifikát */
jarsigner -keystore myKeystore Hello.jar myCert

/* Nakonec umístíme podepsaný applet do kořenového adresáře aplikace s php
    skripty*/
cp ap.jar /var/www/html/virtLab
```

Výpis 2: Instalace Java appletu

Přístupový server a pomocné programy

Zdrojové soubory jsou uloženy v adresáři *src/server*. Soubor **Makefile** obsahuje data pro utilitu **make**.

název	popis
user	jméno pro přístup k db
ps	heslo pro přístup k d
db	název databáz
server_log_file	název souboru pro ukádání aktivit server
dev_tty	název deskriptoru sériového rzhraní

Tabulka 1: Tabulka s proměnnými souboru server.h

Příkazem **make** jsou přeloženy potřebné zdrojové kódy, je vytvořen adresář */usr/virtLab* a nakopírovány do něj všechny spustitelné programy. Do konfiguračního souboru programu Cron */etc/crontab* jsou přidány položky pro automatické spouštění programů *check-timeslot* a *renew-quota*. Nastavení jednotlivých proměnných je definováno v hlavičkovém souboru *server.h*.

```
/* vytvoření serveru, pomocných programů a jejich nakopírování do */
/* adresáře /usr/virtLab, doplnění konfiguračního souboru pro utilitu Cron*/
/* je provedeno voláním programu make v adresáři /src/server. */
make

/* spuštění serveru na portu p */
/usr/virtLab/server p

/* Druhou možností je překlad jednotlivých částí */
*****************************************************************
/* vytvoření serveru,*/
make server

/* kompilace programu renew_quota pro týdenní navyšování kvóty */
make renew_quota

/* kompilace programu check_timeslot pro kontroly začátku timeslotů */
/* začátku nové úlohy */
make check_timeslot

*****************************************************************
/* nastavení utility Cron */
/* do souboru etc/crontab se přidají následující řádky */

0  0 * * 1  virtLab_admin src/server/renew_quota QQ
45 * * * *  virtLab_admin src/server/check_timeslot

/* QQ je parametrem programu říkající, na kolik má být navýšena kvóta */
```

Výpis 3: Instalace serveru a pomocných programů

název	popis
group	skupina pro přístupu do db
user	uživatele pro přístupu do db
passw	heslo pro přístupu do db
db_name	název databáze
server_ip	ip adresa přístupového serveru.
port_server	TCP port přístupvého serveru
com_port_count	nejvetší hodnota čísla konzoly
ulohy_dir	název adresáře s úlohami
zarizeni_dir	název adresáře se zařízeními
tmp_dir	název pomocného adresáře
max_time	max. počet hodin pro rezervaci v jednom dni.
max_session_time	max. doba neaktivity v systému
solution_file	název souboru,s konfiguracemi zařízení
popis_file	určuje název souboru s popisem úlohy
main_xml	název souboru při zadávání úlohy pomocí XML.

Tabulka 2: Tabulka s proměnnými souboru function.php

PHP skripty

PHP zdrojové soubory jsou uloženy v adresáři *src/php/virtLab*, který obsahuje podadresáře **ulohy, zarizeni, tmp**. Jedná se o pomocné adresáře jež aplikace využívá pro ukládání pomocných souborů úloh a zařízení.

PHP skripty je třeba umístit na server. Lze je umístit do libovolného adresáře a nakonfigurovat příslušně Apache (pro ukázku kopírujeme do adresáře */var/www/html/*)

Adresář *src/php/virtLab* nakopírujeme do */var/www/html*.

Soubor **functin.php** obsahuje definice funkcí a proměnných (viz. tabulka 2), které lze v systému nastavit.

Ukázkové úlohy lze do systému vložit skopírovaním adresářů *ukazky/ulohy* a *ukazky/zarizeni* do odpovídajících adresářů na serveru.

B DTD popis úlohy

```
<!--
DTD for Virtual netowork laboratory project
Pavel Nemec
May 2005
-->

<?xml version="1.0"?>
<!DOCTYPE task[
   <!ELEMENT task (id,name,picture_file,timeslots , tbsm_file ,
         description , equipment_list , category_list , link_list )>
   <!ELEMENT id          (#PCDATA) >
   <!ELEMENT name        (#PCDATA) >
   <!ELEMENT picture_file  (#PCDATA) >
   <!ELEMENT timeslots   (#PCDATA) >
   <!ELEMENT tbsm_file   (#PCDATA) >
   <!ELEMENT description (#PCDATA) >
   <!ELEMENT equipment_list(equipment)+ >
   <!ELEMENT equipment  (config)?>
   <!ELEMENT config       (#PCDATA)>
   <!ELEMENT category_list (category)* >
   <!ELEMENT category EMPTY>
   <!ELEMENT link_list  (link)* >
   <!ELEMENT link  (#EMTPY) >

   <!-- type identifies the form of inserted description files -->
   <!ATTLIST desription    type ( single_file |dir|tgz|tar|zip) " single_file " #
       REQUIRED>
   <!ATTLIST equipment    id ID #REQUIRED>

   <!-- attributes of a link -->
   <!ATTLIST link    id_equip1 IDREF #REQUIRED>
   <!ATTLIST link    int1      CDATA #REQUIRED>
   <!ATTLIST link    id_equip2 IDREF #REQUIRED>
   <!ATTLIST link    int2      CDATA #REQUIRED>
   <!ATTLIST equipment id CDATA #REQUIRED>
]>
```

Výpis 4: popis úlohy - DTD

Ukázka vytvoření úlohy pomocí XML

```
<?xml version="1.0"?>
<task>
   <id>OSPF-2</id>
   <name>OSPF protokol</name>
   <picture>ospf2.jpg</picture>
   <timeslots>3</timeslots>
   <dbsm_file>dbsm.conf</dbsm_file>
   <description type="file">descr.tgz</description>
   <seznam_zarizeni>
        <zarizeni id="RA">
     <config>enable
         conf term
          interface eth0
         ip address 192.168.0.1 255.255.255.192
         no shutdown
     </config>
       </zarizeni>
       <zarizeni id="RB">
     <config>enable
         conf term
          interface eth0
         ip address 192.168.0.1 255.255.255.192
         no shutdown
     </config>
       </zarizeni>
       <zarizeni id="RC"/>
       <zarizeni id="SW1"/>
       <zarizeni id="SW2"/>
       <zarizeni id="H1"/>
   </seznam_zarizeni>

   <seznam_linek>
   <linka id_zar1="RA" int1="serial0" id_zar2="RB" int2="serial0"/>
   <linka id_zar1="RA" int1="serial1" id_zar2="RC" int2="serial0"/>
   <linka id_zar1="RB" int1="serial1" id_zar2="RC" int2="serial1"/>
   <linka id_zar1="RA" int1="ethernet0" id_zar2="H1" int2="ethernet0"/>
   </seznam_linek>

   <seznam_kategorii>
      <kategorie id="smerovani"/>
      <kategorie id="jednoduche"/>
      <kategorie id="tps"/>
   </seznam_kategorii>
</task>
```

Výpis 5: Ukázka zadání úlohy pomocí XML souboru

C Seznam souborů

C.1 PHP skripty

1. **administrace_uziv.php**
   administrace uživatelů
2. **ap.jar**
   applet s aplikací terminálu
3. **function.php**
   obecné funkce pro chod webu
4. **index.php**
   hlavní soubor pro přihlašování a odhlašování
5. **init.php**
   provádí inicializaci a kontrolu při přístupech do stránek
6. **interfaces.php**
   stará se o rozhraní
7. **kat_uloh.php**
   práce nad kategoriemi úloh
8. **odstran.php**
   "multifunkční" skript pro odstraňování úloh, zařízení, kategorií, rezervací a nástěnky.
9. **prihlas.php**
   skript pro přihlášení do systému
10. **prochazeni_uloh.php**
    umožní procházení úloh
11. **prochazeni_zarizeni.php**
    umožní procházení zařízení
12. **rezervace.php**
    skript pro zajišťování rezervací
13. **spusteni_ulohy.php**
    skript pro spouštění úlohy
14. **startApplet.php**

15. **statistiky.php**

16. **styl.css**
    CSS styl pro systém

17. **uloz_nastenku.php**
    uložení nástěnky

18. **uloz_rezervaci.php**
    uložení rezervací

19. **uloz_ulohu.php**

20. **vytvor_ulohu.1php**
    první část při ukládání úlohy

21. **vytvor_ulohu.php**
    vytvoření úlohy, umožňuje rovněž editaci úloh a načítání XML dat

22. **xml_function.php**
    Funkce potřebné pro chod XML. Zajišťují obsluhy elementů pomocí SAX API pro PHP.

23. **zarizeni.php**
    správa zařízení, včetně editací.

24. **zpristupneni_ulohy.php**
    zajišťuje funkce pro obsluhu nástěnky.

C.2 Server

1. **server.h**
   hlavičkový soubor pro přístupový server.

2. **renew_quota.c**
   program pro aktualizaci kvóty studentů

3. **Makefile**
   soubor makefile

4. **server.c**
   soubor se zdrojovým kódem přístupvého serveru

C.3 Java applet

1. **TerminalApplet.java**
   Soubor se zdrojovými kódy pro applet terminálu.

2. **MyConnection.java**
   Soubor třídy MyConnection

3. **Protocol.java**
   Soubor s defincemi kódů v hlavičce protokolu.

C.4 Databáze

1. **createDb.sql**
   Skript pro vytvoření databáze, tabulek a uživatele pro přístup k vytvořené databázi

2. **fill_db.sql**
   Soubor s defincemi kódů v hlavičce protokolu.

kód	hodnota	popis
SI	0	si uživatele
AUTH_USER_OK	1	autent. uživatele v pořádku
NOT_AUTH_USER	10	chyba při autentizaci
OK_SI	2	potvrzení přijetí si
NOT_OK	11	nepotvrzeno
ID_ZAR	3	id zařízení
OK_ID_ZAR	4	potvrzení přijetí id zařízení
MAX_LEASE_TIME	5	čas ukončení
OK_MAX_LEASE_TIME	6	potvrzení přijetí
COM_PORT	7	číslo konzoly
OK_COM_PORT	8	potvrzení číslo konzoly
NOT_COM_PORT	12	chybné číslo konzoly
DATA	100	přenášena jsou data
DISCONNECT_NOACTIVITY_1	16	ukončení za 10 min.
DISCONNECT_NOACTIVITY_2	17	ukončení za 5 min.
TIME_EXCEEDED	18	vypršela doba platnosti
ECHO_REQUEST	19	žádost o echo
ECHO_REPLY	20	odpověď na žádost o echo

Tabulka 3: Tabulka s kódy

D Kódy v hlavičce přenášených dat

V tabulce 3 je uveden seznam kódů protokolu včetně jejich významu.