České vysoké učení technické v Praze
Fakulta elektrotechnická

Bakalářská práce

Webové rozhraní pro obecné zabezpečovací zařízení

Jiří Franc

Vedoucí práce: Ing. Kubalík Pavel

Studijní program: Elektronika a informatika strukturovaný bakalářský

Obor: Výpočetní technika

červen 2006

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.
Nemám závažný důvod proti užití tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 25.6. 2006 . . . . . .

Abstract

The goal of this bachelor thesis is to design and produce a web application. This application offers a control of alarms, administration of linked actions, users and alarms. Application communicates with alarms via SMS. Application is able to display a position of subject on the map. Administration of actions, users and alarms is made via database. The important property of the system is its security. This application is a part of a project which consists of a client application, server application and hardware devices.

Abstrakt

Cílem této práce je navrhnout a vytvořit webovou aplikaci, která bude umožňovat řízení alarmů a evidenci událostí s alarmy spojenými. Dále pak správu uživatelů a alarmů. Aplikace bude s alarmy komunikovat prostřednictvím sms zpráv. Jestliže bude alarm umístěn ve vozidle a bude obsahovat GPS modul, aplikace bude zobrazovat získanou polohu na mapě. Evidence událostí, uživatelů a alarmů bude provedena formou databáze. Důraz je kladen na zabezpečení systému proti neoprávněnému přístupu. Tato webová aplikace je součástí projektu obsahujícího klientskou aplikaci, server zajišťující komunikaci a ukládání dat a HW zařízení.

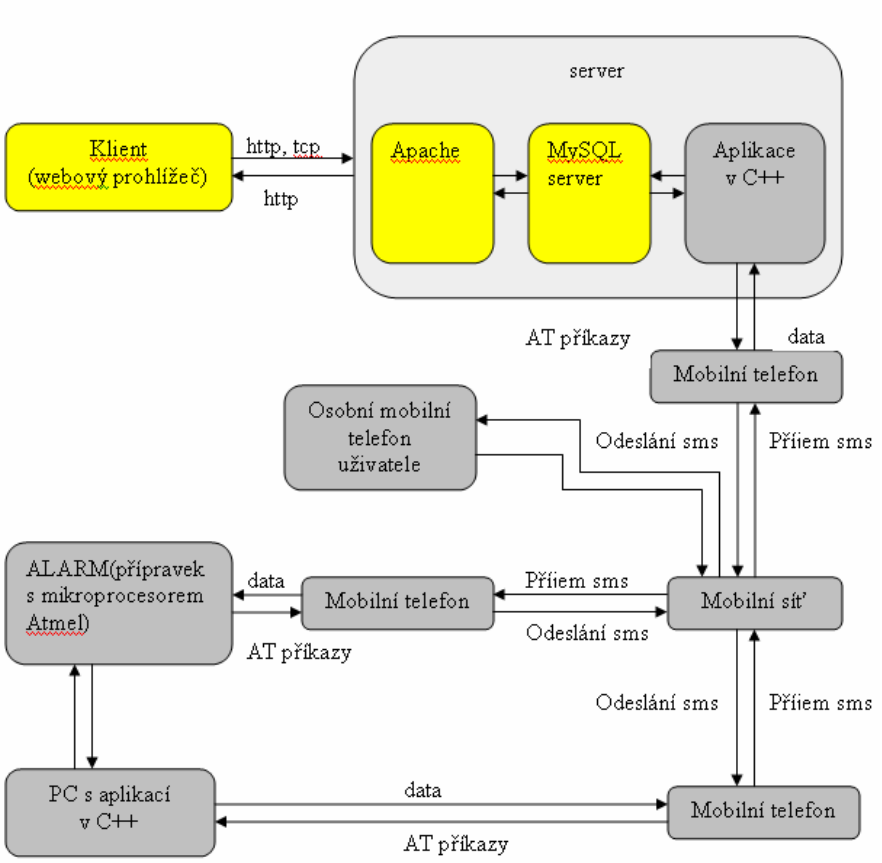
Obsah

Seznam obrázků

Seznam tabulek

Úvod

Cílem této bakalářské práce je navrhnout a realizovat webovou aplikaci sloužící ke správě alarmů. Lze ji rozdělit na dvě části. Část určená uživateli administrátorovi, která bude poskytovat správu nad uživateli a alarmy a část určená obyčejnému uživateli, která bude nabízet správu událostí spojených s alarmy, které obyčejný uživatel vlastní. Z výše uvedeného vyplývá, že aplikace bude podporovat více druhů alarmů. Zmíněné části jsou součástí většího projektu GSM ALARM, jehož topologii můžete vidět na Obr.1. Žlutě označené části jsou tématem této bakalářské práce.



Obr. 1 : Topologie projektu GSM ALARM

Komunikace mezi webovou aplikací a HW přípravkem bude realizována pomocí serverové aplikace, která umožní zaslání sms zprávy prostřednictvím GSM sítě. Pro komunikaci mezi serverem a webovou aplikací bude pak využit protokol TCP, který zajišťuje správné doručení posílané zprávy. Bude třeba vytvořit protokol, který tuto komunikaci zajistí. Jako úložiště dat poslouží databáze umístěná na serveru. Bude obsahovat veškeré informace týkající se uživatelů a jejich alarmů. Součástí této bakalářské práce je její návrh. Přístup do systému bude diferencovaný, což vyplývá z rozdělení na uživatele administrátora a obyčejného uživatele. To se týká také bezpečnosti, protože každý uživatel se přihlásí pomocí logovacího jména a hesla. Toto heslo bude uloženo v zašifrované podobě v databázi. Šifrovat se bude také komunikace prostřednictvím TCP protokolu, aby se zamezilo možnému úniku informací potřebných pro ovládání alarmů. V případě, že bude HW zařízení podporovat

zasílání své polohy, bude možné zobrazit tuto polohu na mapě. K tomu bude využit server www.mapy.cz.

Následující text je koncipován jako návrh softwaru pomocí metod softwarového inženýrství. Je zde úvodní studie obsahující katalog požadavků, kontextový diagram, model jednání, výběr vhodného softwaru a analýza trhu. Další částí je návrh řešení, jehož součástí je návrh datového modelu a jeho popis, funkční model, scénáře použití, dynamický model, návrh akceptačních testů a specifikace základních funkcí. Poslední částí je řešení, ve kterém se zabývám konkrétním programovým řešením projektu. Zbývající kapitoly pojednávají o bezpečnosti systému, o jeho instalaci a testování.

1. Úvodní studie

1.1. Katalog požadavků na funkčnost systému

1.1.1. Základní požadavky

Systém slouží pro správu událostí alarmů obyčejnými uživateli. Z toho plyne, že musí být kladen důraz na bezpečnost aplikace, aby nedošlo k neoprávněnému přístupu do systému. Pokud by k tomu došlo, měla by být možnost vyhledat záznam s informacemi o neoprávněném uživateli. Systém by měl být pro uživatele přehledný a snadno ovladatelný. Stabilita a spolehlivost systému je důležitou součástí dobré funkčnosti. Dostatečná rychlost při komunikaci mezi serverem a webovým rozhraním je potřebná pro včasné informace nabízené uživateli.

1.1.2. Komunikace mezi serverem a webovým rozhraním

Tato komunikace je realizována pouze tehdy, když uživatel chce zaslat příkaz alarmu. Po výběru příkazu jej uživatel odešle a jestliže bude mít nastaveno potvrzování od alarmu, přijde mu potvrzovací odpověď. Odpověď obsahující informativní data dojde, když uživatel zaslal příkaz, který má takovouto odpověď definovanou. Například informace o hodnotě teplotního čidla nebo pozice hlídaného objektu.

1.1.3. Přidávání alarmu

Protože systém podporuje různé druhy hardwarových zařízení (alarmů), je nutné při přidávání alarmu tuto skutečnost nějakým způsobem zohlednit. Obecně má každý alarm nějaký počet vstupů rozdělený na vstupy s číselnou návratovou hodnotou (např. teplotní čidlo, čidlo hluku apod.) a vstupy se stavovou návratovou hodnotou (zapnuto/vypnuto, sepnuto/rozepnuto atd., např. spínací kontakt na dveřích). Každý alarm také obsahuje nějaké výstupy (např. siréna, atd.). Maximální počet těchto vstupů se může lišit právě s ohledem na příslušný alarm. V zásadě bude přidávání alarmu probíhat tak, že se nejdříve vyplní informace o alarmu samotném a jeho typ, poté se odešle tento formulář ke zpracování a poskytne se uživateli další formulář s nabídkou správných počtů vstupů/výstupů, které může přidat. Tento počet může být různý, má však definované maximum, které se odvíjí od typu alarmu. Po zvolení počtu se nabídne uživateli formulář, ve kterém může uživatel zadat jejich jméno, typ a reálnou pozici vstupu/výstupu na HW zařízení.

1.1.4. Přidávání uživatelů

Systém bude mít dva typy uživatelů. Jedním je uživatel administrátor a druhým obyčejný uživatel. Uživatel administrátor bude mít právo přidávat všechny typy uživatelů a obyčejný uživatel jen uživatele se stejným oprávněním. Procedura přidávání probíhá tak, že si uživatel vyplní formulář obsahující informace o novém uživateli, po jeho odeslání se vygeneruje heslo a nabídne se formulář se zmíněným heslem a políčky pro zalogování. Po jeho odeslání se uživatel přidá.

1.1.5. Editace, mazání uživatelů a alarmů

Mazat a editovat uživatele může pouze uživatel administrátor (kromě logovacích údajů, ty může měnit i obyčejný uživatel, pokud jsou jeho). Měl by být tedy k dispozici přehledný výpis, ve kterém lze jednotlivé alarmy a uživatele vyhledávat a potom se dívat na jejich detailní výpis.

1.1.6. Logování přihlášení do systému

Je velice důležitou součástí s ohledem na bezpečnost, jak již bylo zmíněno dříve. Po přihlášení uživatele by se měl vytvořit záznam o této skutečnosti, ze kterého by mělo být viditelné, odkud se uživatel přihlašoval a jaký uživatel se vůbec přihlašoval. Tyto informace budou k dispozici pouze uživateli administrátorovi. Ten bude v těchto záznamech pohodlně vyhledávat a dívat se na statistiky za uplynulý měsíc.

1.1.7. Správa událostí

Obyčejný uživatel bude mít k dispozici přehledný výpis událostí spojených s alarmem. Může tyto události přesouvat do svých složek, případně je mazat. Po kliknutí na ně se zobrazí tabulka vstupů/výstupů, ve které budou barevně odlišené aktivní a neaktivní vstupy/výstupy. K dispozici bude také schéma hlídaného objektu s vyznačenými vstupy opět barevně odlišenými (v případě, že si předtím uživatel přidal schéma hlídaného objektu).

1.1.8. Posílání příkazů

Jak bylo zmíněno dříve, obyčejný uživatel bude zasílat příkaz HW zařízení. Tyto příkazy jsou různé dle typu alarmu, ale lze je rozdělit do několika skupin a to na příkazy pro ovládání, zaslání stavu a nastavení podmínek pro aktivaci alarmu (některé typy alarmů nemusí obsahovat některou ze skupin). Po zvolení skupiny příkazu si uživatel vybere z přehledných formulářů, co chce poslat za příkaz, a potom ho odešle. Podle zvoleného příkazu mu pak dojde odpověď nebo v závislosti na nastavení mu dojde potvrzení o přijetí alarmem.

1.1.9. Plánování akcí

Bylo by vhodné, aby si mohl uživatel naplánovat odeslání zmíněného příkazu. A to jak jednorázové, tak cyklicky opakované odesílání. Například si může navolit zasílání informací o stavu zařízení na každé pondělí v kalendářním roce. K tomuto účelu mu bude sloužit přehledný kalendář, ve kterém budou vepsány zmíněné naplánované akce. Po kliknutí na den se zobrazenou akcí se mu pak vypíše detailní výpis těchto akcí.

1.1.10. Pozice zařízení

Jestliže má obyčejný uživatel přidán zejména alarm určený do automobilu, bylo by vhodné archivovat pozici HW zařízení. Tyto záznamy pak zobrazovat v přehledném výpisu a jednotlivé pozice bude možné zobrazit na mapě.

1.1.11. Nastavení

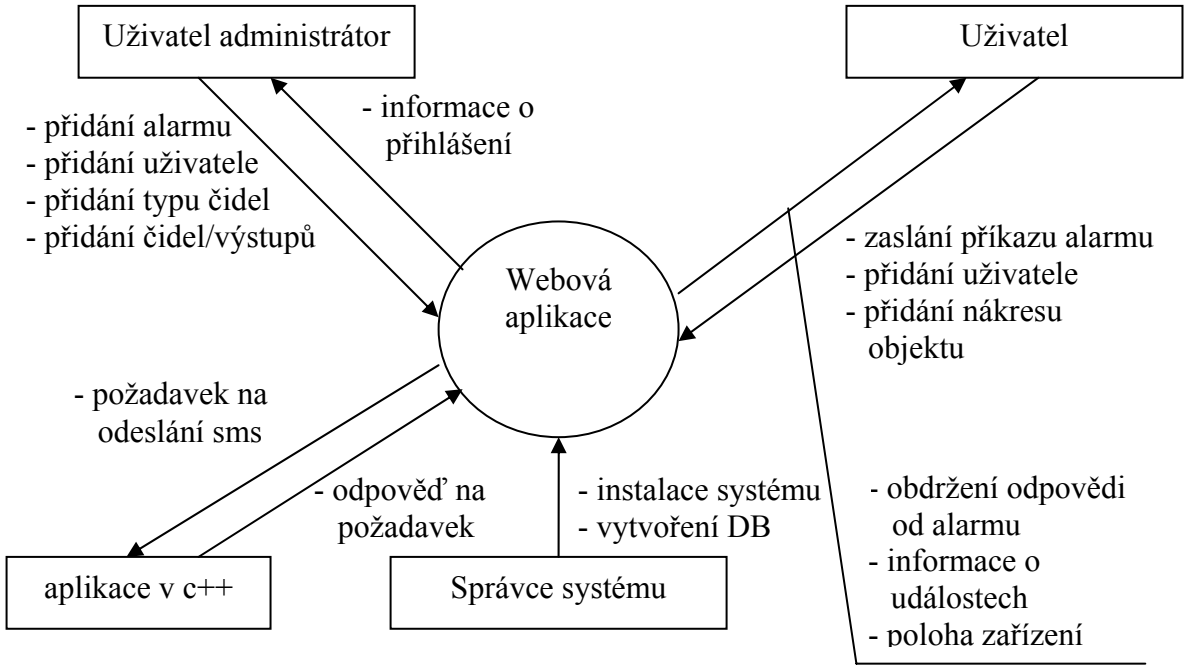
Nastavení je určeno opět obyčejnému uživateli, který může nahrát zmíněné schéma objektu a do tohoto schématu umístit vstupy (čidla) podle reálného umístění těchto čidel. Zde může tato čidla, ale i výstupy pojmenovat, přiřadit jim typ a reálnou pozici, kterou mají na HW zařízení. Bude tu volba, umožňující nastavení odezvy při aktivaci alarmů, například odeslání podrobnější sms nebo emailu. Dříve zmíněné složky, do kterých si může uživatel přesouvat události, se budou přidávat právě zde.

1.1.12. Výběr zařízení

Uživatel může vlastnit více alarmů (např. alarm do budovy a alarm do automobilu) a musí mít tedy možnost si mezi těmito alarmy vybírat.

1.2. Kontextový diagram

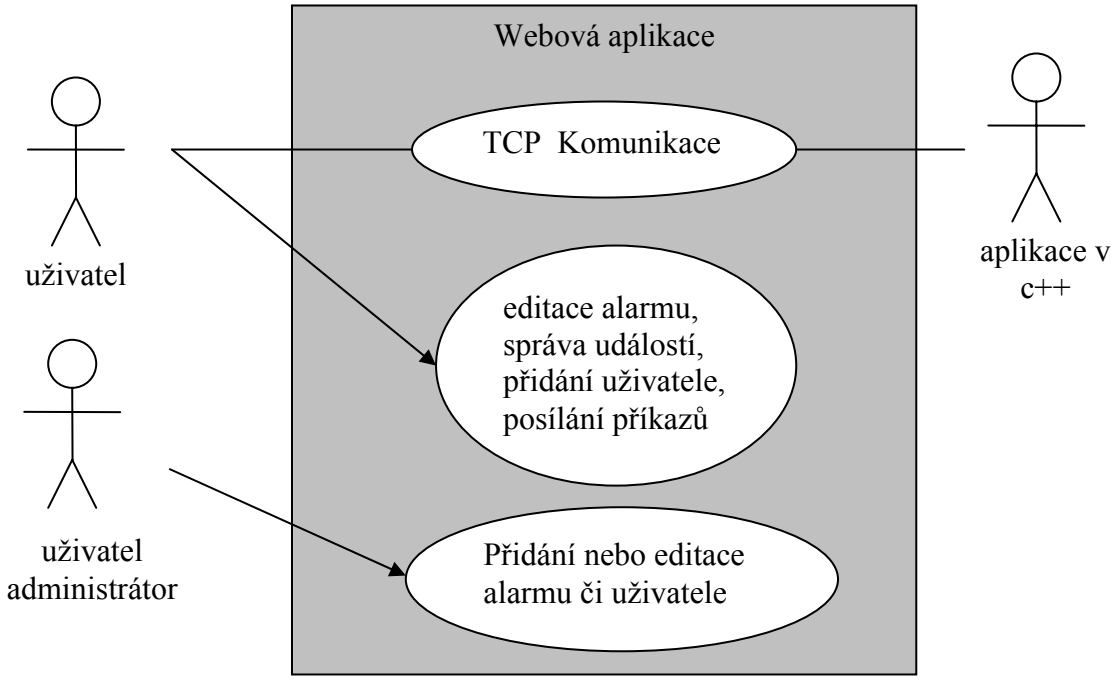
Kontextový diagram zobrazuje aktéry a uživatelské role, které systém obsahuje. Dále zobrazuje datový tok mezi systémem a zmíněnými aktéry. Pro jednoduchost tu nejsou uvedeny veškeré datové toky probíhající v systému, ale jen část těch nejdůležitějších.



Obr. 2 : Kontextový diagram

1.3. Model jednání

Model jednání ukazuje události, na které musí systém reagovat. Jsou zde vidět opět aktéři, které systém obsahuje. Pro jednoduchost jsou uvedeny jen ty nejdůležitější události.



Obr. 3 : Model jednání

1.4. Výběr databáze

Při výběru databáze jsem zohlednil čtyři nejpoužívanější databáze dnešní doby. Z těch komerčních Oracle a DB2 a z OpenSourse databází to jsou MySQL a PostgreSQL. Nejdříve je nutné se zamyslet nad náročností vyvíjené aplikace na databázi. Ta je menší s relativně jednoduchou strukturou. Vystačím se základním, standardním SQL. Počet přístupů do databáze bude omezen počtem registrovaných uživatelů, kterých bude alespoň v prvních třech letech běhu aplikace poměrně málo v porovnání s jinými aplikacemi (např. www.seznam.cz atd…). Důležitou roli hrají také finanční prostředky. Z těchto důvodů odpadají databáze Oracle a DB2. Ze srovnání databází MySQL a PostgreSQL vychází o něco lépe PostgreSQL, která v posledních letech udělala velký krok kupředu. MySQL má však už delší dobu vedoucí postavení na trhu s OpenSource databázemi. Je stabilní a na požadavky aplikace přijatelně rychlá. Její implementace SQL postrádá některé funkce, ale tomuto účelu plně dostačuje. Protože mám s touto databází také předchozí zkušenosti, zvolil jsem tuto. Pro správu vytvořené databáze jsem zvolil OpenSource program phpMyAdmin.

1.5. Výběr programovacího jazyka

Pro tvorbu webové aplikace se nabízí dvě možné varianty. Buď použít technologii aplikačního serveru nebo některý ze skriptovacích jazyků. Pod pojmem aplikační server rozumějte systém, který je napsán například v Javě a má možnost běžet neustále. Jako příklad těchto technologií uvedu J2EE, Java Servlet nebo JSP. Toto je velice dobře použitelné pro

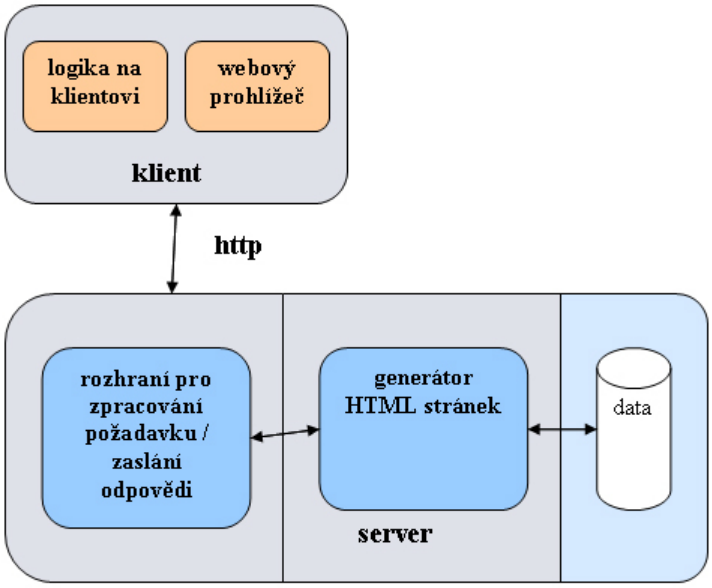
velké webové aplikace, které nutně nemusí obsluhovat jen web, ale i spolupracovat s jinými aplikacemi. Mé webové stránky jsou podstatně menšího rozměru a tato technologie je pro ně zbytečně složitá. Zvolil jsem proto skriptovací jazyk. Při rozhodování, jaký ze skriptovacích jazyků použít, jsou k dispozici tři nejrozšířenější a to PHP, JSP a PERL. JSP není přímo jazyk, ale zkratka pro Active Server Pages. Tato technologie využívá jako programovací jazyk například Visual Basic Script nebo JScript. Její nevýhodou je těžkopádnost, ale také pomalost a nestabilita. Rozdílem mezi Perlem a PHP je, že PHP je od začátku navrženo jako programovací jazyk určený pro web, kdežto Perl je koncipován jako univerzální programovací jazyk. Perl je také daleko těžší integrovat do již vytvořeného HTML kódu. Díky výše uvedeným skutečnostem jsem použil programovací jazyk PHP. Toto rozhodnutí je také dáno mou znalostí tohoto jazyka. Pro tvorbu vlastních webových stránek pak používám HTML 4.01 Transitional a formátovací jazyk CSS 2.

1.6. Výběr softwaru pro tvorbu a běh aplikace

Jelikož aplikaci tvořím na svém osobním počítači, kde je nainstalován operační systém WindowsXP a hotová aplikace bude funkční na počítačích s operačním systémem Linux, zvolil jsem webový server Apache. Pro operační systém WindowsXP jsem použil Apache2Triad, což je balíček obsahující webový server Apache/2.0.54 (Win32) s verzí PHP/5.0.4 a databázový server MySQL s WinMySQLadmin Ver 1.4.

Jako editor pro tvorbu kódu HTML a PHP používám PSPad Freeware editor Ver. 4.5.0 (2183) a pro tvorbu CSS kódu program TopStyle Lite Ver. 3.10.

1.7. Architektura webové aplikace



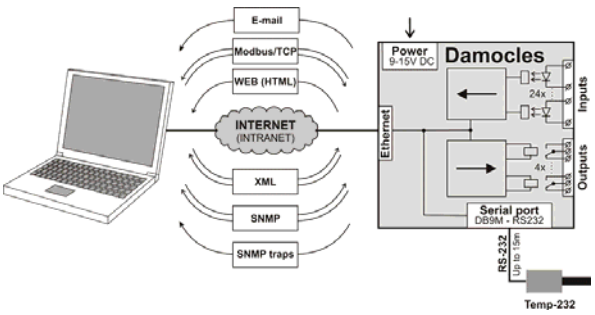
Obr. 4 : Architektura webové aplikace

Logika na straně klienta je psána ve skriptovacím jazyce JavaScript. Slouží k validaci formulářů ještě před odesláním na server. Je také zajímavým nástrojem pro vytvoření dynamických stránek ve smyslu okamžité reakce na nějakou akci. Nevýhodou je však možnost uživatele zakázat si tento jazyk a proto je nutné zajistit funkci stránek i bez

JavaScriptu. Jeho použití se snažím z těchto důvodů omezit. Jazyk pro generování HTML stránek je PHP 5.0.4 a jako úložiště dat slouží MySQL databáze.

1.8. Existující aplikace na trhu

Celkem rozšířené jsou aplikace spojené se sledováním objektů hlavně pomocí webových kamer. Avšak webových aplikací věnovaných přímo alarmu jako takovému je, alespoň na území České republiky, podstatně méně. Začaly se objevovat až v poslední době a jejich počty se pohybují v jednotkách, maximálně desítkách. Jako příklad mohu uvést výrobek Democles firmy HWgroup, ke kterému je nabízena webová aplikace napsaná v XML poskytující informace o čidlech a Flash aplikace, která umožňuje nastavení alarmu. Tento systém má však odlišnou topologii, neboť využívá ke komunikaci mezi HW a serverem přímo internetovou síť, viz obr. XML část zobrazuje stavy binárních vstupů, výstupů a hodnoty naměřené na teplotních čidlech. Tato stránka se každých 5 sekund obnovuje, což umožňuje zobrazení aktuálních hodnot. Flash aplikace pak umožňuje nastavení vstupů, výstupů a teplotních čidel. Aplikace podporuje pouze jeden typ alarmu.
(zdroj: http://www.hwgroup.cz/products/damocles/index_cz.html)

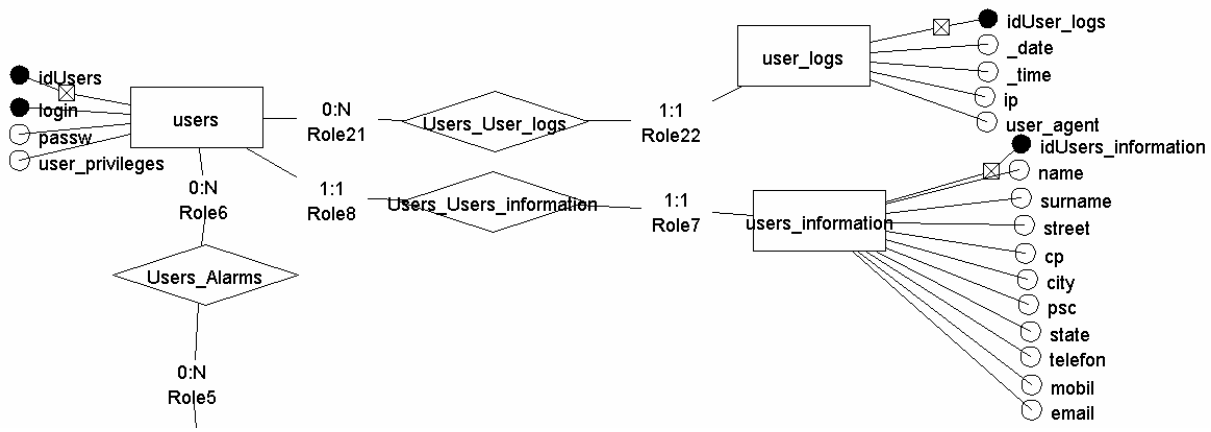


Obr. 5 : Topologie systému Damocles

2. Návrh řešení

2.1. Datový model

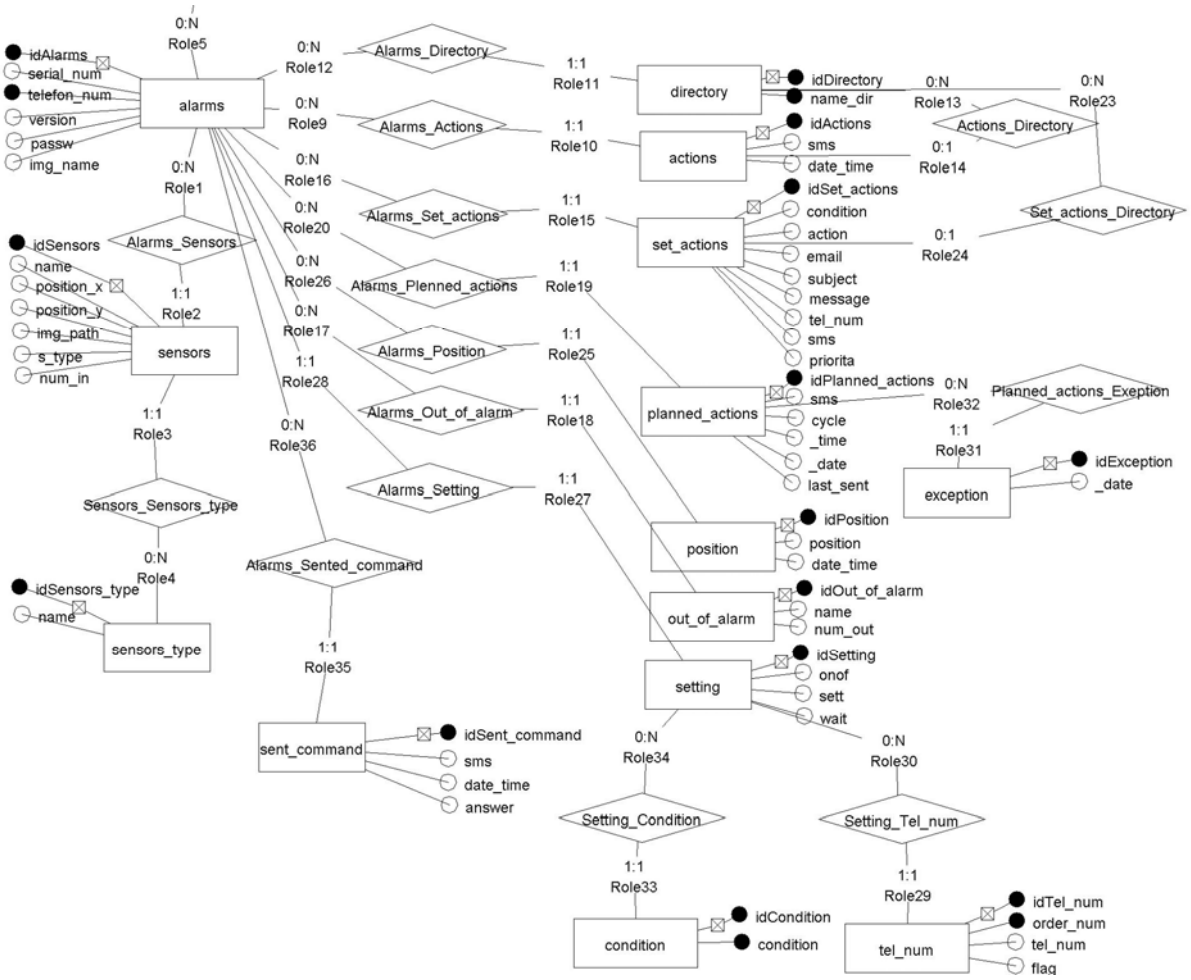
Webová aplikace bude sloužit mimojiné pro evidenci alarmů a událostí spojených s nimi a dále pak pro evidenci uživatelů. Proto jsem zvolil jako prostor pro ukládání těchto dat databázi. Pro návrh databáze jsem použil E-R konceptuální model s využitím programu ER Modeller 3.0 pro Windows32. Navržené entity: users, users_information, user_logs, alarms, sensors, sensors_type, directory, actions, set_actions, pllaned_actions, exception, position, out_of_alarm, setting, sent_command, condition, tel_num. Jejich popis a význam je uveden v kapitole 3.2 . Následující obrázky jsou fragmenty celého E-R modelu, zabývající se jeho dvěmi nejdůležitějšími částmi. Celý E-R model je pak uveden v příloze.



Obr. 6 : E-R model, část věnovaná uživateli

Na tomto obrázku je zobrazena část E-R modelu zabývající se uživatelem. O uživateli jsou ukládány obecné informace a zvlášť informace pro jeho přihlášení do systému.

Obr. 7 se zabývá alarmem, o kterém jsou ukládány informace týkající se jak HW tak softwarové podpory. S alarmem jsou spojeny čidla a výstupy a každé čidlo může mít svůj typ (např. pohybové čidlo). Pro archivaci nastalých událostí slouží entita actions. Události mohou být zobrazovány v různých složkách zadaných uživatelem a uložených v entitě directory. Na vzniklé události se aplikují dané filtry (set_actions). Lze plánovat posílání příkazů (planned_actions) a rušit naplánované (exception). Odeslané příkazy se archivují (sent_command). Ukládá se pozice alarmu. Každý alarm má své nastavení (setting, condition, tel_num).



Obr. 7 : E-R model, část věnovaná alarmu

2.2. Popis datového modelu

2.2.1. Popis jednotlivých entit a jejich atributů

Podrobný popis datového modelu je uveden v příloze.

users

Tato entita je určena pro uložení údajů jednotlivých uživatelů potřebných pro zalogování do systému. Jedním z atributů je také atribut *users_privilages*, který určuje práva uživatele. Ty se striktně dělí na práva uživatele administrátora a práva obyčejného uživatele. Je možné, v případě potřeby, zavést další práva, jako například práva obyčejného uživatele s určeným omezením. Tato hodnota je reprezentována jako řetězec začínající písmenem (v případě admina „a“ a v případě uživatele „u“), za nímž může následovat číselná hodnota určující hladinu oprávnění v rámci těchto dvou tříd.

users_information

Obsahuje podrobnější informace o uživateli. Základem je uživatelovo jméno, příjmení, adresa a emailová adresa. Tyto údaje jsou v přidávání uživatele jako povinné. Dalšími údaji jsou telefonní čísla.

user_logs

Je to entita obsahující informace o přihlášení uživatele do systému. Při jeho přihlášení je nutné uložit čas, datum a ip adresu uživatele. Jako doprovodné informace se ukládají informace o webovém prohlížeči, ze kterého se uživatel přihlásil.

alarms

Obsahuje údaje o jednotlivých alarmech. Protože každý alarm má u sebe připojen mobilní telefon, přes který se komunikuje, je třeba si uložit jeho číslo. Pro ochranu této komunikace se pak využívá heslo, které je uloženo v atributu *passwd*. Jako každý HW výrobek obsahuje i alarm sériové číslo a verzi programu, který je do něj nahrán, nebo verzi HW. A protože je každý alarm umístěn v nějakém objektu, který lze vyjádřit náčrtem, obsahuje entita alarm atribut *img_name*, ve kterém je uložena cesta ke zmíněnému náčrtu.

sensore

V této entitě jsou uloženy informace o čidlech připojených k danému alarmu. Uživatel si může každé čidlo pojmenovat, tedy atribut *name*. Důležitým atributem je *type*, který určuje typ čidla. Čidlo může být buď spínací, vracející hodnotu nula nebo jedna, respektive sepnuto/rozepnuto, nebo čidlo s číselnou návratovou hodnotou, jako například teplotní čidlo vracející aktuální teplotu. Tyto dva typy pak reprezentuje řetězec, který má pro spínací čidlo hodnotu „s“ a pro čidlo s návratovou hodnotou „h“. U čidel je nutné si pamatovat jejich reálné připojení na HW přípravek, aby bylo možné zasílat příkazy se správnou syntaxí. K tomu slouží atribut *num_in*, který obsahuje číslo daného portu. Jelikož lze do náčrtu alarmu rozmisťovat čidla, je nutné si pamatovat jejich pozici v rámci tohoto náčrtu (atribut *position_x*, *position_y*) a obrázek, který čidlo reprezentuje (atribut *img_path*).

sensors_type

Tato entita určuje možné typy čidel. Znamená to, že se čidla dělí do určitých tříd, které mají něco společného. Jako příklad lze uvést třídu s názvem pohybová čidla (atribut *name*).

directory

Uživatel si může nastalé události uložit do složek, které si sám vytvoří. Dané události se ve skutečnosti přiřadí cizí klíč složky. Složka má jméno – atribut *name_dir*

actions

Jestliže HW přípravek spustí alarm nebo odešle odpověď na příkaz, ukládají se akce v rámci této entity. Akce přichází v textové formě pomocí sms. Sms je pak uložena v atributu *sms*. Je ukládán čas a datum přijetí sms.

set_actions

Přijatou sms od alarmu (událost) lze filtrovat a provádět na základě nastavení různé akce. Atribut *condition* pak určuje provedení akce. Nabývá číselných hodnot, které reprezentují zvolenou podmínku. Akce, která se bude provádět, je uložena v atributu *action*. Ta může nabývat celkem tří hodnot: 1 – přesun do složky, 10 – odeslání emailu, 100 – zaslání sms na

uložené telefonní číslo. Jsou možné kombinace těchto hodnot. Zde je jejich výčet: 1,10,11,100,101,110,111. Jednotlivým filtrům lze přiřadit prioritu, ve které se budou provádět (atribut priorita). Pro odeslání emailu jsou pak logicky zapotřebí atributy *email*, *massage* (obsah emailu), *subjekt* (předmět emailu). Pro odeslání sms pak *tel_num* (telefonní číslo) a *sms* (text zprávy).

planned_actions

Systém umožňuje plánovat zasílání příkazů alarmům. K tomu slouží tato entita. Je třeba si pamatovat příkaz, který se bude posílat (atribut *sms*) a čas s datem, kdy se příkaz odešle (atribut *_time*, *_date*). Aplikace umožňuje provádět cyklické posílání příkazů, což znamená zasílat příkaz například každé pondělí v osm hodin. K tomu slouží atribut *cycle*. Jeho obsah se skládá z roku, měsíce, týdne, dne. Například 2004000102, což znamená, že vybraný příkaz se odešle v roce 2004, každý měsíc v prvním týdnu a v druhý den týdne, tedy v úterý. Jestliže není akce cyklická, je nastaven tento atribut na nulu.

exception

Tato entita je pomocná k předešlé. Slouží pro uložení výjimek cyklických akcí. Jestliže uživatel nechce smazat celou cyklickou akci, ale jen v jeden den a hodinu udělat výjimku a tuto akci neprovést, uloží se výjimka právě do tabulky *exception*. K identifikaci výjimky slouží datum (atribut *_date*).

position

Alarmem může být i zařízení umístěné v automobilu a podporující GPS technologii. Zasílá informace o své poloze. Ty se pak ukládají spolu s datem do tabulky *position*.

out_of_alarm

K alarmu jsou připojené výstupy, které se aktivují na základě vstupů. Jako například siréna nebo spínač, který odstartuje případnou akci. Uživatel si může jednotlivé výstupy pojmenovat (atribut *name*) a musí jim přiřadit číslo portu, ke kterému jsou připojeny na HW přípravku.

setting

U HW přípravku lze nastavit některé jeho defaultní vlastnosti. Typicky, zda je vypnutý nebo zapnutý (atribut *onof*) a zda bude odesílat potvrzovací sms na příchozí příkaz (atribut *wait*). Každý HW má také další nastavení, které se liší od ostatních. Toto nastavení se pak ukládá do atributu *sett* v podobě řetězce. Řetězec vypadá u alarmu home 1 takto: *TEMP<param>,OUTP<param>,INPU<param>,SPIN<param>*. Obecně se pak tento řetězec skládá z příkazů a jejich parametrů, přičemž jsou jednotlivé příkazy odděleny čárkami.

condition

U každého alarmu se mohou nastavit určité podmínky, při kterých se aktivuje výstup. Podmínky vlastně reprezentují sepnutí jednotlivých vstupů nebo jinou akci související se vstupy. V této tabulce jsou uloženy nastavené podmínky. Ukládají se do atributu *condition* ve tvaru, v jakém jsou zasílány HW zařízení při nastavování. Například u alarm home 1 vypadá tato podmínka následovně: *SETR<param>*.

tel_num

U alarmu lze nastavovat oprávněná telefonní čísla, aby nešlo alarm ovládat z jakéhokoliv telefonního čísla. Tato čísla jsou pak uložena v tabulce *tel_num*. Vlastní telefonní číslo je

v atributu *tel_num*. Atribut *flag* určuje, zda z příslušného čísla lze zasílat příkazy pro nastavení (hodnota atributu: s) nebo se na příslušné číslo budou zasílat informativní sms (hodnota atributu: r). Telefonní čísla jsou v HW zařízení uložena v pořadí, které je důležité hlavně při zasílání informativních sms. Prvnímu telefonnímu číslu se sms odešle jako první. Zde je toto pořadí reprezentováno atributem *order_num*.

sent_command

Uživatel by měl mít možnost si zpětně zobrazit příkazy, které již zaslal HW přípravku. Ty se ukládají do této tabulky. Ukládá se datum a čas (atribut *date_time*) odeslání příkazu a jeho obsah (atribut *sms*).

users_has_alarm

Tabulka sloužící k realizaci spojení entit *alarms* a *users*, které mají mezi sebou relaci M:N.

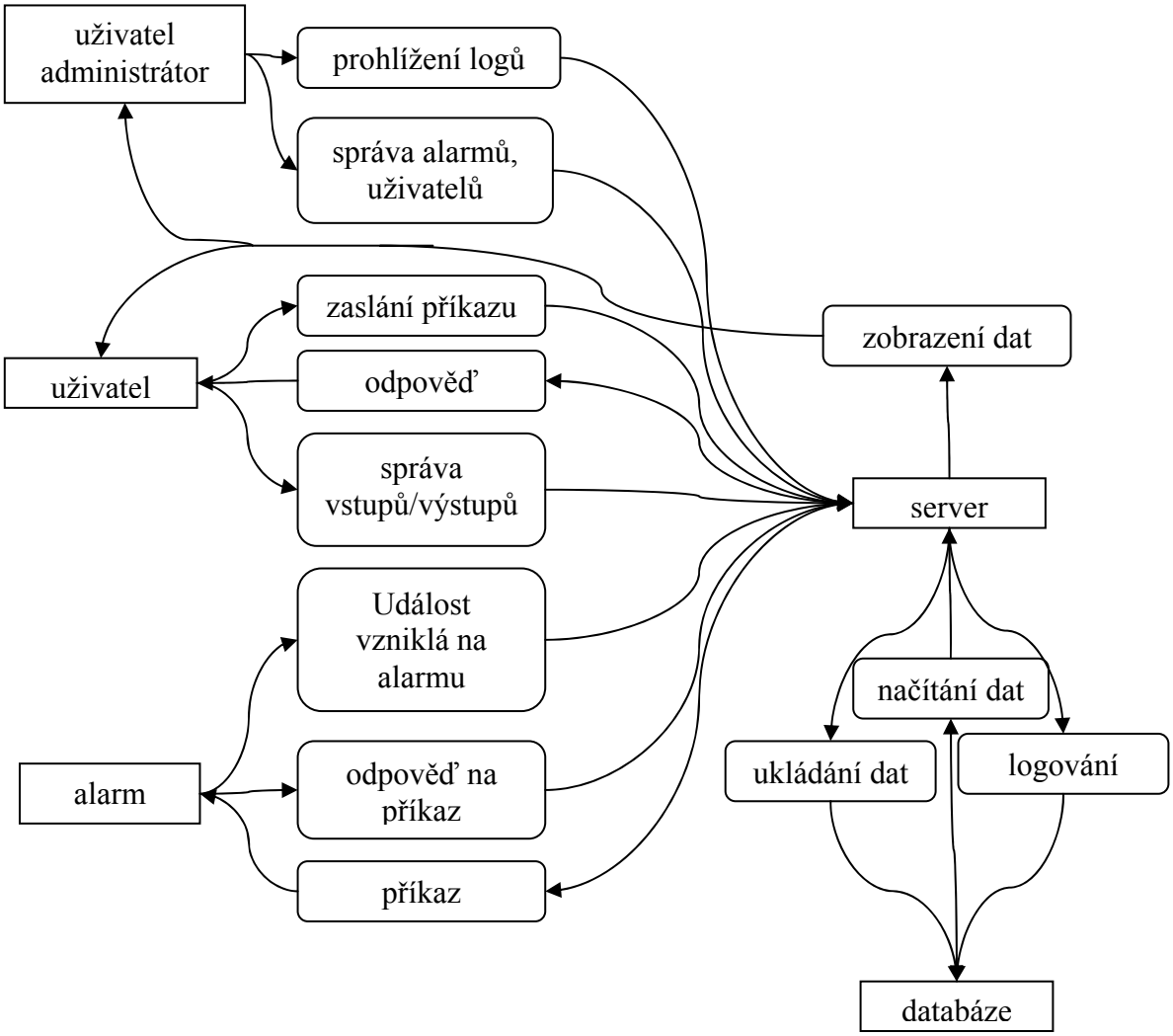
2.2.2. Relace mezi entitami

Relace Users – Users_iformation
Typ realce: 1 – 1
Jeden uživatel musí mít právě jeden záznam v entitě *Users* svázaný právě s jedním záznamem v entitě *Users_information*.
Relace Users – User_logs
Typ realce: 1 – N
Jeden uživatel se může 0 až N x přihlásit do systému a tím vytvořit log a jeden log je vázán právě na jednoho uživatele.
Relace Users - Alarms
Typ relace: N – M
Jeden alarm může obsluhovat 0 až N uživatelů (např. členů rodiny). Naopak jeden uživatel může mít k dispozici 1 až N alarmů (např. alarm v domě, autě, chatě…).
Relace Alarms - Sensors
Typ relace: 1 – N
Alarm může mít u sebe připojeno 0 až N senzorů a daný senzor je připojen právě k jednomu alarmu. Počet senzorů je prozatím omezen na 8 spínacích a 4 teplotní.
Relace Sensors – Sensors_type
Typ relace: 1 – N
Senzor musí být právě jednoho typu. Naopak jeden typ může zahrnovat 0 až N senzorů.
Relace Alarms – Directory
Typ relace: 1 – N
Pod jedním alarmem může být uloženo 0 až N složek a daná složka je vždy vázána právě k jednomu alarmu.
Relace Alarms - Actions
Typ relace: 1 – N
Alarm může signalizovat 0 až N událostí a daná událost je vždy vázána právě k jednomu alarmu.
Relace Alarms – Set_actions
Typ relace: 1 – N
Alarm může aplikovat na příchozí zprávy 0 až N filtrů a daný filtr je vždy vázán právě k jednomu alarmu.

Relace Alarms – Plenned_actions
Typ relace: 1 – N
Alarm může odeslat 0 až N uložených sms a daná sms je vždy vázána právě k jednomu alarmu.
Relace Alarms – Position
Typ relace: 1 – N
Alarm může být postupně na 0 až N pozicích a daná pozice je vždy vázána právě k jednomu alarmu.
Relace Alarms – Out_of_alarm
Typ relace: 1 – N
Alarm může mít u sebe připojeno 0 až N výstupů a daný výstup je připojen právě k jednomu alarmu.
Relace Alarms – Setting
Typ relace: 1 – 1
Alarm může mít právě jedno nastavení a jedno nastavení patří k právě jednomu alarmu.
Relace Alarms – Sent_command
Typ relace: 0 – N
V rámci jednoho alarmu se může odeslat 0 až N příkazů a tyto příkazy patří vždy právě k tomuto alarmu.
Relace Actions – Directory
Typ relace: 1 – N
Jedna událost může být v 0 až 1 složce a jedna složka může obsahovat 0 až N událostí.
Relace Set_actions – Directory
Typ relace: 1 – N
V Jednom filtru může být nastaven přesun do 0 až 1 složky a přesun do jedné složky může být definován u 0 až N filtrů.
Relace Plenned_actions - Exception
Typ relace: 1 – N
Cyklická plánovaná akce může mít 0 až N výjimek, kdy se neprovede, a daná výjimka je vždy svázána právě k jedné cyklické akci.
Relace Setting – Condition
Typ relace: 1 – N
Nastavení alarmu může obsahovat 0 až N podmínek a jedna podmínka je vždy svázána s právě jedním nastavením alarmu.
Relace Setting – Tel_num
Typ relace: 1 – N
Nastavení alarmu může obsahovat 0 až N telefonních čísel a jedno telefonní číslo je vždy svázáno s právě jedním nastavením alarmu.

2.3. Funkční model

Funkční model zobrazuje interakci mezi jednotlivými objekty. Pro jednoduchost jsou opět vybrány pouze ty nejdůležitější.



Obr. 8 : Funkční model

Poznámka

Interakce odpověď mezi uživatelem a serverem označuje odpověď na zasílaný příkaz.

2.4. Scénáře použití

Scénáře použití se dělí na dvě základní části a to část administrátorskou a uživatelskou. Uživatelem je zákazník, který si zakoupil alarm.

2.4.1. Uživatel administrátor

- Přidání alarmu

V případě zakoupení HW zařízení obyčejným uživatelem uživatel administrátor uloží údaje o tomto zařízení prostřednictvím formuláře do databáze. V rámci této procedury přidá k danému alarmu základní počet senzorů a případně je pojmenuje.

- Editace, mazání alarmu
- Vyhledávání alarmu, řazení alarmů
- Přidání, editace, mazání typu čidel
- Registrace uživatelů

  Administrátor nejprve vyplní formulář s informacemi o uživateli a vybere mu příslušný alarm. Po odeslání těchto informací se administrátorovi zobrazí formulář s vyplněnými logovacími údaji, které může pozměnit. Vybere práva budoucího uživatele a stiskne tlačítko „save“. Tímto je registrace ukončena.
- Výpis, editace uživatelů

  Administrátor může vyhledat informace o uživateli a pozměnit je nebo mu přidat/odebrat alarmy a změnit jeho logovací informace.
- Kontrola logů

  Slouží pro kontrolu přístupů do systému. V logech lze vyhledávat a administrátorovi je k dispozici několik možných výpisů logů dle různých pravidel. Lze zobrazit i kalendář s logy, z kterého je dobře vidět frekvence přihlašování uživatelů.

2.4.2. Obyčejný uživatel

- Volba alarmu

  Protože může uživatel mít více alarmů, musí si v první řadě vybrat alarm, se kterým hodlá pracovat.
- Výpis událostí

  Uživatel si může vyhledat danou událost a podívat se na podrobnosti kolem ní. Danou událost pak může přesunout do ním definované složky událostí, kde s ní lze nakládat stejným způsobem.
- Plánování akcí

  Zde uživatel plánuje akce, které se budou pravidelně opakovat nebo se provedou jednorázově. Například zasílání informační sms každé první pondělí v měsíci nebo jen jednou v daný čas a datum. V případě, že akce není již potřebná, může ji uživatel zrušit.
- Posílání příkazů

  Uživatel může zasílat příkazy alarmu, které jsou různého typu. A to řídící, informační a nastavovací. V případě, že má uživatel aktivován alarm na odesílání potvrzení o přijetí, aplikace vyčká na tuto odpověď a zobrazí ji. Po úspěšném odeslání se příkaz uloží do DB a uživatel si pak může prohlížet odeslané příkazy. Další možností je zobrazení aktuálního nastavení alarmu, které je závislé také na odesílaných příkazech.
- Zobrazení pozice alarmu

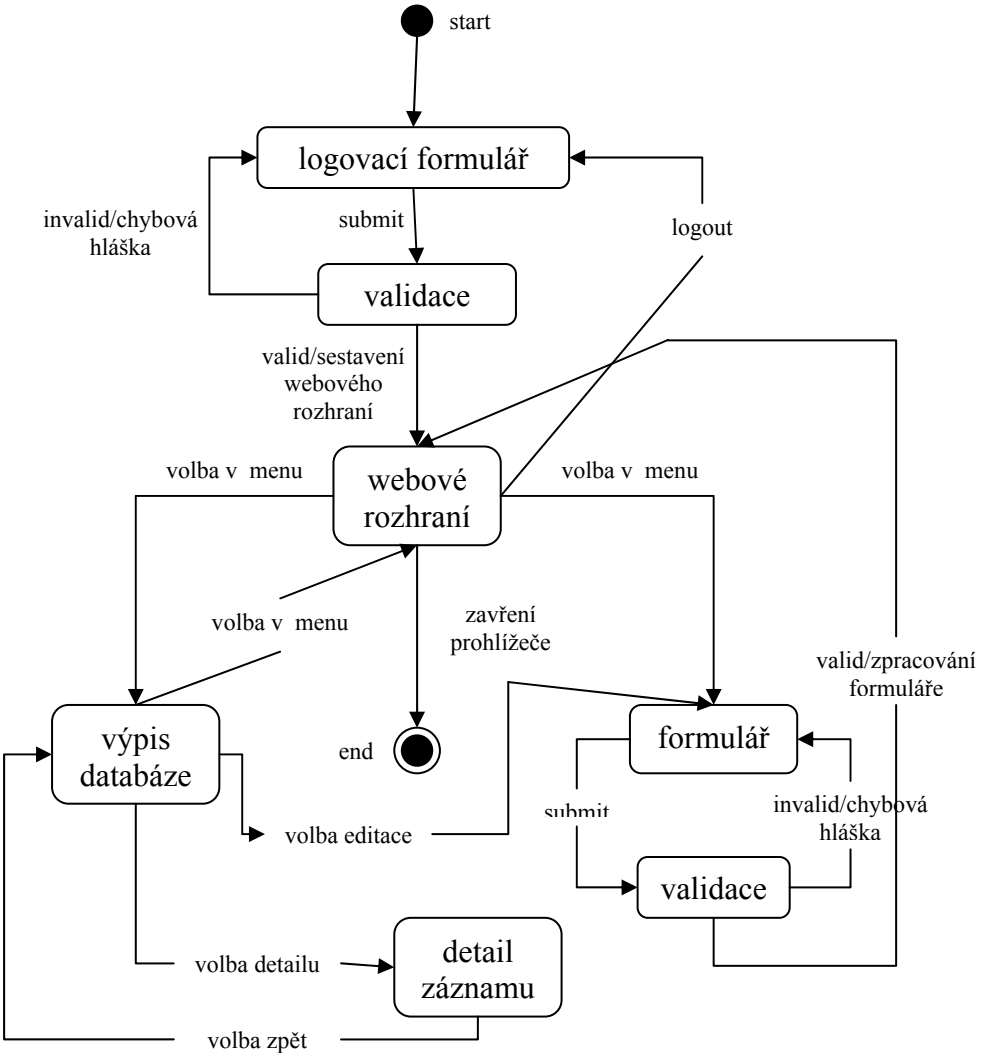
  V případě zaslání příkazu pro zjištění polohy se přijatá informace o pozici uloží do DB a může ji zobrazit ve výpis nebo na mapě.
- Nastavení

  V nastavení jsou k dispozici informace o alarmu a o pojmenování čidel a výstupů. V případě připojení nových čidel nebo aktivaci nových výstupů u alarmu je zde formulář pro přidání čidel nebo výstupů. Uživatel si může uložit

schéma (např. půdorys) objektu, ve kterém je alarm umístěn a poté do něj vložit čidla podle skutečnosti. V případě aktivace alarmu si může uživatel nastavit reakci na tuto skutečnost. Má na výběr z přemístění události do jiné složky nebo odeslání emailu nebo odeslání sms. Zmíněné složky může přidávat a mazat. Uživatel také může registrovat nového uživatele, např. rodinného příslušníka.

2.5. Dynamický model

Dynamický model popisuje stavy aplikace, ve kterých se může nacházet. Na modelu jsou vidět jen zobecněné stavy pro lepší pochopení aplikace. Obecně se dá říci, že vám aplikace nabízí buď formulář pro editaci dat nebo výpis dat, ze kterého se lze dostat dále na editaci nebo detailní výpis. Počáteční stav *start* je vlastně zadání URL adresy do některého z webových prohlížečů. Koncový stav *end* je pak zavření tohoto prohlížeče nebo změna URL nebo jiné ukončení nebo selhání spojení uživatele se serverem.



Obr. 9 : Dynamický model

2.6. Návrh akceptačních testů

Sada otázek, na které existuje odpověď pouze ano/ne a které vypovídají o tom, zda jsou splněny požadavky zadavatele.

- Může obyčejný uživatel zaslat příkaz pouze svému alarmu?
- Jsou obyčejnému uživateli k dispozici všechny události vzniklé na alarmu, které se mají posílat na server?
- Může uživatel přidat jiného uživatele (příslušníka rodiny) pouze k alarmům, které mu patří?
- Vidí uživatel administrátor opravdu všechna přihlášení uživatelů, která byla vykonána?
- Je možné přidat dva alarmy se stejným číslem?
- Jsou správně nastaveny maximální počty vstupů/výstupů, které lze přidat k danému alarmu?

2.7. Specifikace základních funkcí

Systém obsahuje typicky jeden model funkce pro výpis formulářů a pro výpis záznamů databáze. Funkce pro výpis má většinou jako vstupní parametry dvě pole, která obsahují názvy sloupců, které se budou vypisovat, a nadpis těchto sloupců. Dalšími parametry může být proměnná page a action, které slouží pro řízení aplikace. Funkce pro výpis formuláře obsahuje daný formulář a jako parametry má typicky proměnné page a action. Dalším typem je funkce pro verifikaci hodnot získaných z formulářů. Ta má návratovou hodnotu buď true nebo false v závislosti na výsledku verifikace.

3. Řešení

3.1. Konfigurace aplikace

Z důvodu předpokladu instalace aplikace na různé servery musí aplikace obsahovat konfigurační soubor. V tomto souboru jsou uloženy proměnné, které jsou závislé právě na atributech serveru a pro jednoduchost jsou zde uloženy i informace o databázi. Gramatika pro tento soubor je následující:

```
SET       -> ATRIBUT = HODN endl SET
SET       -> epsilon
ATRIBUT -> db_server
ATRIBUT -> db_username
ATRIBUT -> db_password
ATRIBUT -> db_name
ATRIBUT -> tcp_ip
ATRIBUT -> tcp_port
ATRIBUT -> tcp_timeout
HODN    -> string
```

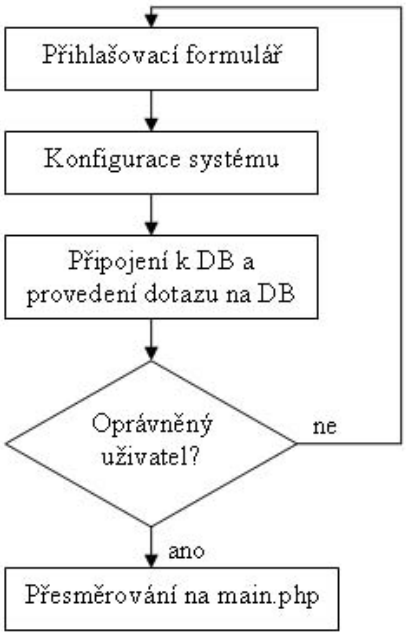
Tab. 1 *Gramatika INI souboru*

Proměnná db_server obsahuje název serveru, na který se bude databáze připojovat, tedy typicky localhost (127.0.0.1). db_username a db_password je přihlašovací jméno a heslo pro připojení k databázi a ab_name je jméno databáze. Proměnné tcp_ip, tcp_port, tcp_timeout slouží, jak vyplývá z jejich názvu, pro nastavení tcp spojení mezi aplikací a serverem. Tcp_ip je tedy ip adresa serveru, tcp_port je port serveru, přes který se bude komunikovat a tcp_timeout je čas, po který aplikace čeká na odpověď. Tato komunikace bude blíže popsána v kapitole 3.11. Komunikace se serverem. Všechny tyto hodnoty se během přihlašování uživatele ukládají do proměnných SESSION, které jsou uloženy na straně serveru a jsou k dispozici po dobu, po kterou je uživatel přihlášen.

3.2. Přihlašování do systému

Autentizace uživatele probíhá pomocí kontroly dat v databázi a uložení těchto dat do proměnných SESSION. Po zadání internetové adresy uživatelem (ta může být různá, s ohledem na server, na kterém aplikace běží) se zobrazí přihlašovací formulář. Po jeho odeslání se zadané hodnoty zpracovávají skriptem login.php. Zde se nejdříve nakonfiguruje systém (nastaví se proměnné uvedené v kapitole 3.1. Konfigurace aplikace), poté se skript připojí k databázi a provede dotaz, zda existuje uživatel se zadaným přihlašovacím jménem a heslem. V případě úspěchu se uloží informace o přihlášení uživatele do tabulky user_logs (proměnné jsou detailně popsané v Popisu datového modelu) a nastaví se proměnné $_SESSION['user_id'], $_SESSION['entry_session'], $_SESSION['user_role'].

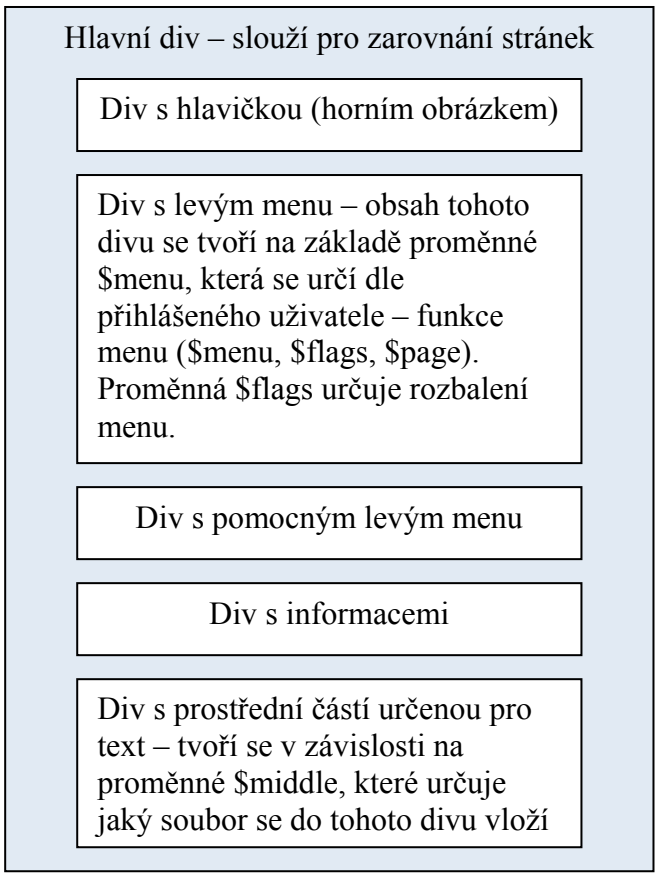
Dále se pomocí funkce headers přesměruje na skript main.php. V případě neúspěchu se vypíše oznámení o špatném zadání jména a hesla a aplikace nabídne opět přihlašovací formulář.



Obr. 10 : Vývojový diagram – přihlašování do systému

3.3. Logické rozdělení HTML stránky

HTML stránka je rozdělena do pěti základních částí. První z nich je hlavička stránky, která plní informativní a estetickou úlohu. Druhou částí je menu aplikace, třetí je pak pomocné menu. Čtvrtá část je informativní a poslední částí je střed aplikace obsahující zobrazované texty a tabulky. Toto rozdělení je provedeno pomocí HTML tagu div. Všechny zmíněné části jsou pak umístěny v jednom hlavním divu (viz obr 4.4). Ve skriptu divs.php je pak toto rozdělení implementováno.



Obr. 11 : Logické rozdělení stránky pomocí divů.

3.4. Sestavení a zobrazení HTML stránky

3.4.1. Řídící skript main.php

O sestavení a zobrazení stránky se stará skript main.php. Jestliže je proměnná $_SESSION ['entry_session'] nastavená na 1 (uživatel je řádně přihlášen), rozhodne se mezi částí pro administrátora a obyčejného uživatele podle proměnné $_SESSION ['user_role']. V obou částech se nastavují proměnné $middle a $menu v závislosti na proměnné $page. Proměnná $middle obsahuje řetězec s názvem skriptu, který generuje střed stránky, a proměnná $menu obsahuje pole, ve kterém je uloženo dané menu. Po jejich nastavení se vypíše hlavička HTML stránky, obsah podle proměnných $middle, $menu a ukončení HTML stránky. V případě prvního generování stránky (tedy hned po přihlášení) pro obyčejného uživatele se místo uvedeného postupu vloží skript first_choose.php sloužící k výběru alarmu. V případě odhlášení uživatele se odstraní existující proměnné SESSION, vypíše se informativní zpráva o uživatelově odhlášení a zobrazí se přihlašovací formulář. Když uživatel zadá jakoukoliv jinou adresu stránek než index.php a není přihlášen, zobrazí se stránka, jež ho informuje o časovém vypršení platnosti stránek.

3.4.2. Generování menu stránky a informativní části

Levé menu je generováno ze dvou typů dvourozměrných polí. První pole obsahuje položky 1. úrovně menu (jeho struktura je uvedena níže). Druhé pole obsahuje položky 2. úrovně menu a jeho struktura je následující: Výpis pole je realizován funkcí menu, jejímiž parametry jsou název pole 1. úrovně, flag, má-li se daná položka rozbalit a daná položka, na kterou bylo kliknuto. Funkce obsahuje cyklus, ve kterém prochází pole 1. úrovně a tiskne jednotlivé položky. Když se narazí na položku, která má 3. hodnotu v poli rovnu 3. parametru funkce a je nastaven flag na 1, toto podmenu se vypíše.

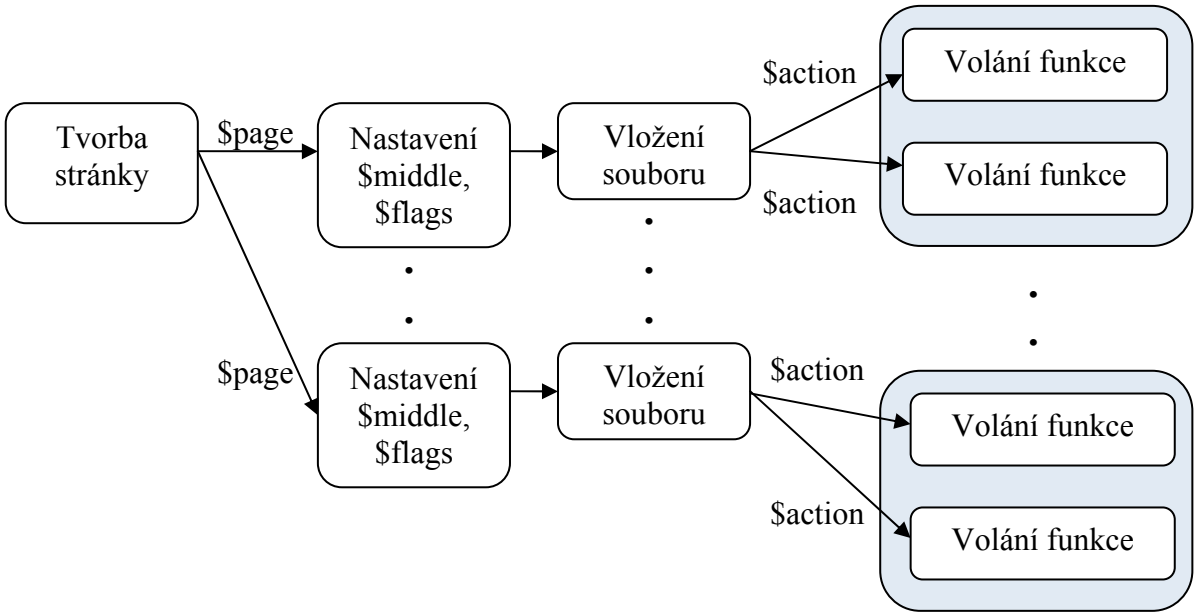
Struktura pole menu:
$nazev = array ("0" => array ("vypisovaný text", "odkaz (obsah parametru href)", "název pole s hodnotami menu 2.úrovně"))
Struktura pole submenu:
$nazev = array ("0" => array("vypisovaný text", "odkaz (obsah parametru href))

V pomocném menu je položka logout a help, která je statická a společná pro admina a pro obyčejného uživatele.

V informativní části je vypsán obsah proměnné $_SESSION ['left_div']. V případě obyčejného uživatele je tato hodnota naplněna informacemi o alarmu, který je zvolen. Jestliže je proměnná prázdná, informativní div se nezobrazuje.

3.4.3. Generování prostřední části stránky

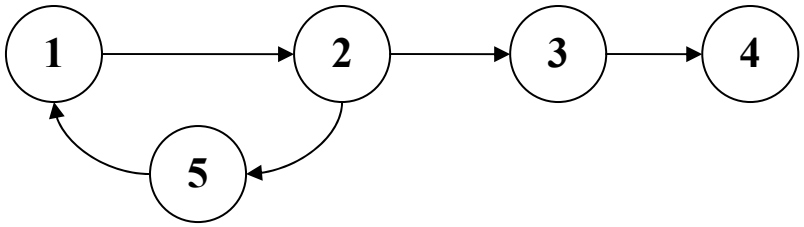
K vytvoření středové části slouží proměnné $page, $action, $action_in. Pomocí proměnné $page se určuje obsah proměnných $middle a $flags zmíněných výše. Po nastavení těchto hodnot se vloží do středové části odpovídající soubor. V rámci tohoto souboru rozhoduje proměnná $action o tom, jaká se zavolá funkce nebo provede akce, a jestliže je třeba, tak pod ní rozhoduje proměnná $action_in (není již uvedena na obrázku). Viz následující obrázek.



Obr. 12 : Generování prostřední části stránky

3.5. Zpracování formulářů

Po vyplnění formuláře uživatelem se tento formulář odešle ke kontrole. Zde se vyhodnotí správnost zadaných údajů a poté se buď formulář dále zpracovává nebo se vypíší chyby, které udělal uživatel při vyplňování formuláře a zobrazí se mu tento formulář.



Obr. 13 : Stavový automat zpracování formuláře

1 – výpis formuláře
2 – kontrola formuláře
3 – zpracování formuláře
4 – výpis výsledků
5 – výpis chybových hlášení

Tab. 2 *Popis stavů automatu*

3.6. Výpis záznamů

Výpis záznamů tabulek, pokud je realizován, je prováděn do tabulky, která má následující rozložení. V hlavičce tabulky je její název, případně rozsah (např. 0 až 15) vypisovaných záznamů a celkový počet záznamů. Na druhém řádku hlavičky jsou vypsány názvy sloupců, které odpovídají vybraným sloupcům tabulky databáze. Pokud jsou implementovány v rámci tabulky filtry, jsou tyto názvy provedeny jako odkazy (popsáno níže kapitola 3.7. Filtry). Dalším řádkem hlavičky (v případě použití filtrů) jsou inputboxy, sloužící pro filtr hledání. Tělo tabulky obsahuje samotná data. Levé sloupce (0 až 3 v závislosti na vypisované tabulce) slouží jako řídící. Obsahují ikonu pro mazání záznamu, zobrazení editace záznamu nebo pro zobrazení detailu záznamu. Do zbylých sloupců se vypisují jednotlivé atributy záznamu. Jeden řádek je tedy jeden záznam tabulky.

3.7. Filtry

V rámci výpisu záznamů jsou implementovány dva základní filtry pro řazení a hledání záznamů. Kvůli těmto filtrům je potřeba si vždy pamatovat SQL dotaz, který byl naposledy vykonán. Ten je uložen v proměnné $_SESSION [‘query’]. Z důvodu bezpečnosti je opět zvolena pro uložení proměnná SESSION.

Filtr pro řazení je realizován jako nadpis sloupce, který je odkazem s parametry určujícími danou akci. Skript, který vyhodnocuje kliknutí na tento odkaz, projde předchozí

SQL dotaz a zjistí, zda v něm není část pro řazení. Když ano, vyhodnotí, zda neobsahuje ten samý parametr pro řazení, podle kterého se řadilo v předchozím kroku. Jestliže ano, změní hodnotu SQL dotazu DESC na ASC nebo naopak. Jestliže se jedná o nový parametr pro řazení, vymění pouze ten. Protože tento skript upravuje pouze část SQL dotazu (ORDER BY <parametr> {DESC, ASC}), lze ho kombinovat i s ostatními filtry.

Hledání je realizováno pomocí inputboxů, kde se zadá požadovaná část hledaného řetězce u daného sloupce a poté se tlačítkem search odešle ke zpracování. Příslušný skript analyzuje předcházející SQL dotaz, který si rozdělí na tři části (jestliže se v něm vyskytují). Jsou to části následující po WHERE, část ORDER BY a část LIMIT. Poté vytvoří svou vlastní část WHERE z přijatých inputboxů, které jsou v SQL dotazu napojovány logickou funkcí AND. Nakonec připojí předešlou část WHERE opět pomocí funkce AND, část ORDER BY a poslední část LIMIT.

3.8. Funkční prvky

Jako funkční prvky jsou uvedeny ikony pro editaci, detail a mazání záznamů v levých sloupcích tabulky. Dalším funkčním prvkem je tzv. stránkování výpisu záznamu, kdy se zobrazuje pouze určitá část záznamů (v této aplikaci 15).

Ikony pro mazání, editaci a detail jsou realizovány jako odkazy obsahující parametry určující skript, který je bude zpracovávat, tedy hodnoty proměnných action, page, případně action_in. Dále pak parametry potřebné pro toto zpracování. Typicky primární klíč daného záznamu.

Pro stránkování výpisu záznamů se musí nejdříve nastavit maximální nebo také celkový počet záznamů. To se provádí SQL dotazem SELECT s příslušnými parametry na databázi. Samotné stránkování je prováděno částí SQL dotazu LIMIT, kde je třeba dvou hodnot. Tyto hodnoty reprezentují rozsah vybíraných záznamů, respektive první hodnota udává od kolikátého záznamu se bude vybírat a druhá, kolik záznamů se vybere. První hodnota je uložena v proměnné $_SESSION ['limit_down'], druhá je implicitně nastavena na 15. Jestliže uživatel klikne na některé z tlačítek *šipka doleva, doprava*, přičte se nebo odečte od proměnné limit_down 15 s ohledem na celkový počet záznamů a na nulu. Dvojitá šipka doleva/doprava slouží pro rychlý přesun na začátek/konec. V případě, že máme méně záznamů než 15 nebo, že jsme na začátku/konci, příslušné šipky se nezobrazí.

Obr. 14 : Ukázka výpisu záznamů

3.9. Část pro administrátora

3.9.1. Přidání alarmu

Aplikace má podporovat různé typy alarmů, proto je třeba přidávání alarmu tomuto přizpůsobit. Všechny přidávané alarmy mají společné atributy jako sériové číslo, verze, telefonní číslo, důležitým atributem je typ alarmu, který právě odpovídá různým druhům alarmů. Tyto informace se uloží a následuje přidání čidel a výstupů, jejichž počet je závislý na druhu alarmu. Obecně jsou rozlišovány u alarmu dva druhy čidel: čidlo spínací, které vrací hodnotu zapnuto/vypnuto a čidlo s návratovou hodnotou, které vrací určitou číselnou hodnotu (například teplotní čidlo vracející číselný údaj o teplotě). Výstupy jsou pak u všech alarmů reprezentovány stejně. Po přidání alarmu je zavolána funkce pro přidání čidel/výstupů s parametry odpovídajícími počtu čidel/výstupů k danému alarmu. Podle tohoto rozdělení alarmů se pak řídí část určená obyčejnému uživateli, protože každý alarm podporuje částečně jiné příkazy.

3.9.2. Registrace nového uživatele

Tato procedura obsahuje dva kroky. Nejprve uživatel (administrátor) vyplní podrobné informace o novém uživateli typu jméno, příjmení, adresa atd. a poté, jestliže tyto informace vyplnil správně, se uloží do mezivýsledku. Administrátorovi je nabídnut formulář pro vyplnění uživatelova loginu, hesla a práv. Až po správném vyplnění se teprve uloží login, heslo a práva do tabulky users a informace do tabulky users_information. Heslo je do databáze ukládáno jako hash spojený ze dvou částí. První z nich je heslo uživatele a druhé z nich je náhodně zvolené (je pevně zvoleno v kódu). Toto je prováděno z důvodu bezpečnosti, protože je pro útočníka mnohem těžší najít z takového hashe jeho obraz (což souvisí s prolomením hashovací funkce md5). Jako hashovací funkci je používána haval256.

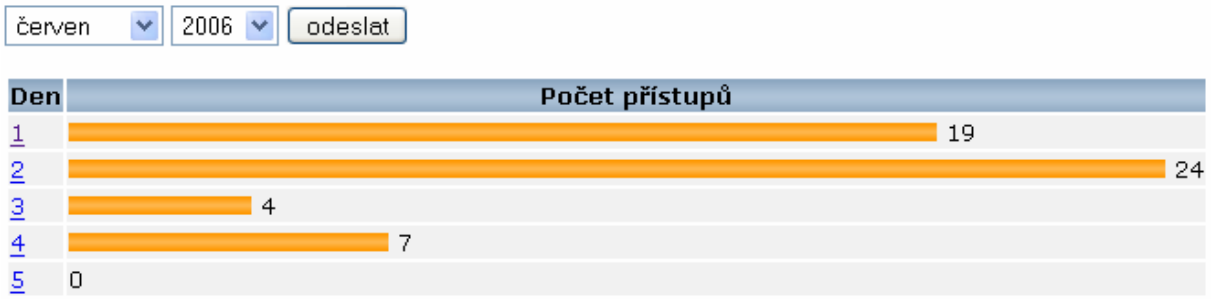
3.9.3. Statistiky

V této části si může administrátor prohlédnout datum, čas a jiné informace o přihlášení uživatele. Je zde vyhledávání, které nabízí hledat podle jména, příjmení, alarmu, práv uživatele a data přihlášení. Po odeslání tohoto formuláře se zavolá skript, který generuje SQL dotaz dle zadaných informací. Dále se zobrazí výsledek do tabulky. Při ukládání logu uživatele při jeho přihlášení se ukládají mimo jeho ip adresy a času s datem přihlášení i informace získané z http hlaviček, které poskytuje globální proměnná prostředí s názvem SERVER. Ukládá se hodnota ip adresy uživatele REMOTE_ADDR a prohlížeč, ve kterém uživatel otevřel tuto webovou aplikaci HTTP_USER_AGENT.

Detail logu	
Datum přihlášení	2006-06-01
Čas přihlášení	19:13:28
IP adresa uživatele	127.0.0.1
Prohlížeč uživatele	Mozilla/5.0 (Windows; U; Windows NT 5.1; cs; rv:1.8.0.3) Gecko/20060426 Firefox/1.5.0.3
Informace o uživateli	
Jméno	Jan
Přijmení	Rychloš
Adresa	Novotného 15,Praha,12356,CZ
Telefon	605869568
Mobil	256982123
Email	janrychli@gmail.com

Obr. 15 : Ukázka detailu záznamu

Další zajímavou částí je kalendář, kde se uživateli (administrátorovi) zobrazí jím požadovaný měsíc v daném roce. Uživatel zde může vidět graficky znázorněnou návštěvnost stránek v jednom měsíci. Každý řádek odpovídá jednomu dni a datum tohoto dne je pak odkazem na detailní výpis uživatelů přihlášených v tento den. Výpis vybraného měsíce probíhá v cyklu od 1 do počtu dnů měsíce a v každém průchodu se provede SQL dotaz na tabulku user_logs, zda obsahuje nějaké záznamy s příslušným datem. Jestliže ano, zjistí se, kolik těchto záznamů je a vykreslí se obrázek, jehož šířka je upravena vztahem (int)(($count*600)/$max_length). Proměnná count odpovídá počtu přihlášených uživatelů v daném dni a proměnná max_length odpovídá maximu přihlášených uživatelů za den v daném měsíci. Číslo 600 je pak šířka kalendáře. Jestliže v daný měsíc nebyl nikdo přihlášen, je proměnná max_length nastavena na 1. Tímto způsobem se pak zobrazí graf, na kterém je velice dobře vidět poměr mezi jednotlivými počty přihlášených uživatelů v různých dnech.



Obr. 16 : Ukázka statistiky zobrazené v kalendáři

3.10. Část pro obyčejného uživatele

3.10.1. Události

Události evidované v rámci alarmu jsou zobrazovány do tabulky, která byla popsána již dříve, s tím rozdílem, že každý prvek má checkbox, kterým ho lze vybrat a provést požadovanou akci. Ta je umístěna na konci tabulky. Vybrané události lze buď smazat nebo

přesunout do jiné složky. Přesun do jiné složky je realizován editací atributu directory_idDirectory entity action na hodnotu odpovídající vybrané složce. V událostech jsou zobrazeny jen ty události, které nemají přiřazenou žádnou složku. Při otevření detailu události se zobrazí čidla a výstupy a ty, které jsou aktivní, se zvýrazní. Jestliže má uživatel přidán nákres hlídaného objektu (popsáno dále), zobrazí se mu i se zvýrazněnými aktivními čidly.

3.10.2. Moje složky

Tato položka menu je určena právě pro složky uživatele. Po kliknutí na ni se rozbalí a zobrazí se uživateli jeho složky, které mají stejnou funkčnost jako výše popisovaná položka menu události, ale zobrazují se jen události, které mají nastaven atribut directory_idDirectory právě na jejich hodnotu.

3.10.3. Plánované akce

Plánovanou akcí se rozumí naplánování poslání příkazu pomocí mobilního telefonu. Uživateli jsou k dispozici dva typy plánovaných akcí. Jedním je jednorázová plánovaná akce a druhým cyklická akce. Obě jsou zobrazovány do stejného kalendáře. Ten je realizován pomocí tabulky, kdy se pro žádaný měsíc nejdříve zjistí den, ve kterém měsíc začíná, aby se mohl vypsat první řádek kalendáře. Potom se v každém dni provede dotaz na databázi, zda neobsahuje plánovanou akci s tímto datem nebo zda není nastavena cyklická akce s parametry odpovídajícími tomuto datu (popsáno níže). Spočítá se celkový počet těchto akcí a v případě, že je větší než nula, vypíše se do daného dne ikona detailu a daný počet akcí. Tato ikona je odkazem na detail dne, kde se může uživatel podívat na detailní výpis zadaných akcí. V databázi je však akce uložena v podobě, jakou se bude posílat alarmu, proto je nečitelná pro normálního uživatele a je třeba ji nějakým způsob převést. O tento převod se stará třída parse_sms, která hledá v uložené sms příkazy a ukládá jejich textový ekvivalent do pole, které následně vrací aplikaci. Přidávání jednorázových akcí je realizováno přes kalendář vypsaný jako v předchozím případě. Každý den je odkaz, kterým si uživatel vybírá datum, potom si zadá čas a vybere z nabídky příkazů. Akce je pak uložena. Přidávání cyklických akcí probíhá přes formulář, kde si uživatel vybere periodu opakování jeho příkazu. Příkaz je možné provádět v daný měsíc nebo každý měsíc, daný týden nebo každý týden, v daný den nebo každý den a v daný rok. Tato hodnota je v DB uložena v tomto formátu 2004040102, kdy první čtyřčíslí udává rok, následující dvojčíslí měsíc, další týden a další den. Jestliže je zadáno například každý měsíc, tak příslušné dvojčíslí je nastaveno na 00. Zbylá část přidávání akce je stejná jako u akce jednorázové. Zajímavostí je mazání cyklické akce, protože bylo třeba, aby mohl uživatel smazat akci pouze v jeden časový interval a ne celou. Takovéto smazání je provedeno tak, že se do pomocné tabulky exception uloží ID, čas a datum cyklické akce. Při celkovém mazání této cyklické akce se pak odstraní i tento záznam.

Obr. 17 : Ukázka kalendáře s naplánovanými akcemi

3.10.4. Posílání příkazů

Posílané příkazy jsou různé pro různé alarmy. Podle vybraného alarmu se vloží odpovídající části kódu. Posílání příkazů je vlastně implementace protokolu PC-AVR a dalších. Uživatel si vybere typ příkazů (typem se rozumí ovládání, stav atd.), který chce odeslat, nastaví příslušné parametry příkazu a pak jej odešle. O poskládání reálného příkazu v podobě odpovídající protokolu PC-AVR a dalších se stará třída check_sms. Obecně se vytvoří objekt této třídy, v jehož konstruktoru se nastaví ID alarmu, kterému se příkaz posílá, a proměnná určující, zda alarm odešle potvrzovací sms. Poté se zavolá zvolená funkce nad tímto objektem podle typu zvolených příkazů. Té se předají parametry získané z formuláře, které se uloží do třídních proměnných. Z této funkce je zavolána funkce pro tvorbu samotné sms. Ta obecně obsahuje volání funkcí jednotlivých příkazů, ale vždy má na začátku volání funkce Get_head, ve které se vygeneruje první část sms obsahující telefonní číslo a heslo alarmu a na konci volání funkce Get_end, která přidá na konec příkazů ENDC a parametr určující zaslání odpovědi od alarmu. Hlavnímu programu aplikace je pak předán string obsahující sms připravenou k odeslání. Samotné odesílání sms je realizováno prostřednictvím serveru, kterému je tato sms doručena přes TCP. Tato komunikace je popsána v kapitole 3.11. Komunikace se serverem.

3.10.5. Pozice zařízení

Zde je uživateli k dispozici výpis pozic hlídaného objektu. Zobrazení pozice na mapě je provedeno prostřednictvím webového portálu www.mapy.cz. Mapy.cz používají geografické souřadnice (zeměpisná šířka/zeměpisná délka) na elipsoidu a datu WGS-84. Pro zobrazení pozice musí uživatel kliknout na tuto pozici, která je odkazem právě na portál Mapy.cz. Tento odkaz má následující strukturu:

http://mapy.cz/?st=search&fr=loc:50°30’0.00’’N%2015°00’00’’E

Řetězec 50°30’0.00’’N 15°00’00’’E reprezentuje geografickou polohu o souřadnicích 50°30’0.00’’N 15°00’00’’E. Protože nejvyšší rozlišení map je v této chvíli 50cm/pixel, pohybuje se teoretická maximální geodetická přesnost v řádu jednotek setin vteřiny.

3.10.6. Nastavení

Uživatel si může přidávat a editovat čidla podle svého. Tento algoritmus je stejný jako u administrátora. Aby měl uživatel k dispozici nákres svého objektu, musí ho nahrát na server. To probíhá tak, že si uživatel pomocí formuláře vybere daný nákres, ten se zkontroluje, zda je typu jpg a zda má odpovídající rozměry a velikost. Uloží se do adresáře users_img ve tvaru <*číslice*>.jpg a poté se upraví atribut img_name v entitě alarms na hodnotu odpovídající číslici v názvu. Při ukládání se nejdříve zjistí, zda nemá alarm již nějaký nákres přidán, když ano, nový nákres ten starý přepíše, pokud ne, najde se nejvyšší možné číslo reprezentující název nákresu, inkrementuje se a nákres se uloží pod tímto jménem. Umístění čidel do obrázku se dělá pomocí HTML objektu inputu typu image, který vrací pozici x, y bodu, ve kterém bylo na něj kliknuto. Umístění čidla probíhá tak, že si uživatel vybere čidlo a obrázek, který ho bude reprezentovat a klikne do nákresu na místo, kam chce čidlo umístit. Tím se odešle tento formulář spolu s již zmíněnou pozicí kliknutí. Tato pozice je uložena do databáze k příslušnému čidlu. Vykreslení čidel do nákresu je pak provedeno přes pozicování, kdy nákres je umístěn v HTML objektu div, jež má nastaven vlastnost position na absolute, do něho jsou pak umístěny obrázky jednotlivých čidel opět pomocí objektu div, který má nastavenu vlastnost position také na absolute a navíc má nastavenu vlastnost left, top na hodnoty x, y z DB, jež jsou hodnoty zvolené uživatelem v předchozím bodě. Dále lze nastavit reakci na přijatou událost jako odeslání emailu, sms nebo přesouvat tyto události rovnou do některých složek. V nastavení lze také přidávat a mazat své složky. Uživatel může registrovat nového uživatele, který může mít přidány pouze alarmy, které má přidány původní uživatel. Algoritmus pro registraci nového uživatele je totožný s algoritmem popsaným v části pro administrátora.

3.10.7. Výběr zařízení

Během přihlašovací procedury si musí uživatel zvolit alarm, s kterým bude pracovat. ID tohoto alarmu je uloženo v proměnné $_SESSION[‘idAlarms’] a ta se dále využívá v ostatních algoritmech. Tato volba pak slouží pro změnu alarmu, do zmíněné proměnné se uloží jiné ID.

3.11. Komunikace se serverem

Tato komunikace slouží pro předávání dat mezi skripty PHP (server apache) a aplikací napsanou v c++. Je vytvořena z důvodu posílání sms a přijímání odpovědí na ně na webovém prostředí prostřednictvím mobilního telefonu připojeného k serveru a obsluhovaného právě aplikací v c++. Komunikace je vytvořena pomocí protokolu pro řízení přenosu TCP (Transmission Kontrol Protokol), který je definován v RFC 905 a slouží pro výměnu dat mezi klientem a serverem. K realizaci propojení je třeba znát adresu serveru, tj. IP adresu a číslo portu služby, která je v tomto případě nastavena při konfiguraci systému.

3.11.1. Klient

Zasílá následující příkazy:

- Odeslání sms příkazu alarmu:
  send_sms(<*tel. číslo*>,"<*sms*>",<*parametr*>)
  tel. číslo podoba telefonního čísla je <předvolba><číslo>, tedy například 420607589126
  sms sms je text, jehož obsah se řídí protokolem PC – GSM

parametr Vyjadřuje, zda bude klient čekat na odpověď serveru nebo ne. Tato hodnota je závislá na nastavení alarmu, které je uloženo v DB.

HODNOTA	POPIS
W	klient čeká na odpověď
N	klient nečeká na odpověď

Tab. 3 *Parametr – možné hodnoty*

Po zaslání tohoto příkazu klient čeká na potvrzení příjmu příkazu serverem. Doba čekání je určena hodnotou proměnné $_SESSION['tcp_timeout']. Čekání probíhá v cyklu, který má nastaveno 10 průchodů. Aplikace čeká na odpověď 10 x tcp_timeout. V případě, že aplikace nedostala potvrzení, oznámí uživateli, že příkaz nebyl odeslán. Naopak jestliže potvrzení dojde, aplikace se rozhoduje, zda bude čekat na odpověď alarmu nebo ne a to dle parametru, který je v posílaném příkazu (n,w). V případě, že aplikace bude čekat, proběhne stejná procedura jako při čekání na první potvrzení. Když proběhne přenos v pořádku, aplikace zašle serveru příkaz quit(), kterým ukončí komunikaci.

- Příkaz pro ukončení komunikace:
  quit()

3.11.2. Server

Zasílá následující příkazy:

- Potvrzení o přijetí příkazu od klienta.
  ack(<*number*>)
  numer číslo reprezentující pozici příkazu ve frontě obsahující sms k odeslání

- Odpověď přijatá serverem od alarmu
  deliverd_sms("<*sms*>")
  sms sms je text, jehož obsah se řídí protokolem PC – GSM

3.12. Reprezentace dat

Typografická konvence

Název entity	
Atrubut 1	datový typ atributu 1
.	.
.	.
Atrubut N	datový typ atributu N

PK – primární klíč, int
FK – cizí klíč, int

users	
idUsers	PK
login	varchar(128)
passw	varchar(128)
users_privilages	varchar(2)

users_information	
idUsers_information	PK
Users_idUsers	FK
name	varchar(30)
surname	varchar(30)
street	varchar(100)
cp	varchar(10)
city	varchar(45)
psc	varchar(20)
state	varchar(45)
telefon	varchar(20)
mobil	varchar(20)
email	varchar(100)

directory	
idDirectory	PK
Alarms_idAlarms	FK
name_dir	varchar(15)

setting	
idSetting	PK
Alarms_idAlarms	FK
onof	char
wait	char
sett	varchar(200)

alarms	
idAlarms	PK
seriove_c	varchar(45)
telefon_c	varchar(20)
verze	varchar(45)
passw	varchar(8)
img_name	int

sensors_type	
idSensors_type	PK
name	varchar(100)

user_logs	
idUser_logs	PK
user_id	FK
_date	DATE
_time	TIME
ip	varchar(30)
user_agent	TEXT

actions	
idActions	PK
Alarms_idAlarms	FK
sms	varchar(255)
date_time	DATETIME

set_actions	
idSet_actions	PK
Alarms_idAlarms	FK
direktory_idDirectory	FK
condition	int
action	int
email	varchar(100)
subjekt	varchar(100)
message	TEXT
tel_num	varchar(20)
sms	varchar(255)
priorita	int

position	
idPosition	PK
Alarms_idAlarms	FK
position	varchar(20)
date_time	DATETIME

out_of_alarm	
idOut_of_alarm	PK
Alarms_idAlarms	FK
name	varchar(100)
num_out	int

sent_command	
idSent_command	PK
Alarms_idAlarms	FK
sms	varchar(200)
date_time	DATETIME

condition	
idCondition	PK
setting_idSetting	FK
condition	varchar(50)

users_has_alarm	
idUsers_has_alarm	PK
Alarms_idAlarms	FK
Users_idUsers	FK

tel_num	
idTel_num	PK
setting_idSetting	FK
order_num	int
tel_num	varchar(20)
flag	char

planned_actions	
idPlanned_actions	PK
Alarms_idAlarms	FK
sms	varchar(255)
cycle	int
_time	TIME
_date	DATE
last_sent	TIME

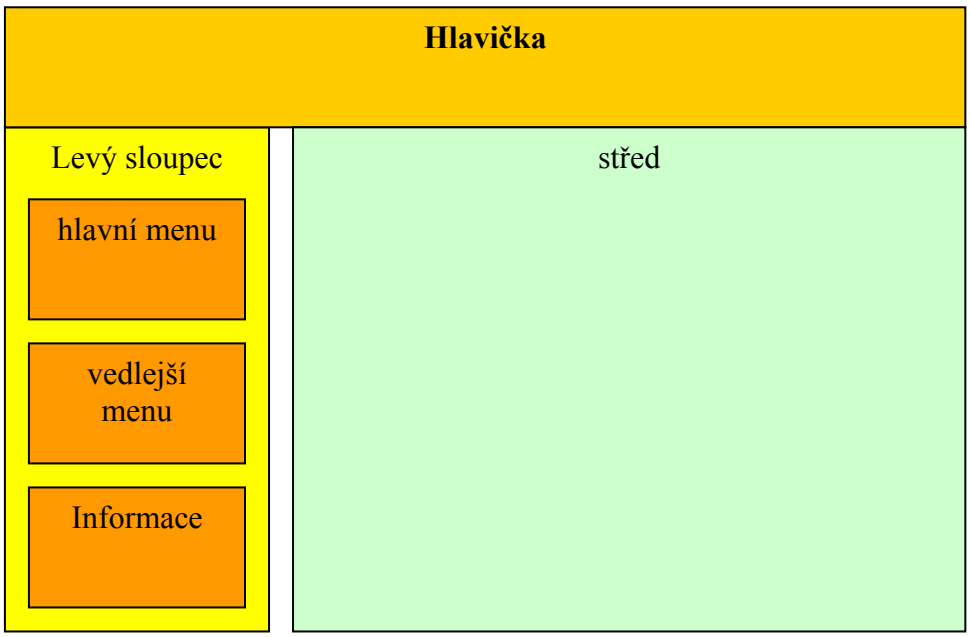
sensore	
idSensors	PK
Alarms_idAlarms	FK
sensore_type_idSensors_type	FK
name	varchar(100)
positron_x	int
positron_y	int
img_path	varchar(30)
type	char(1)
num_in	int

exception	
idException	PK
planned_actions_idPlanned_actions	FK
_date	DATE

3.13. GUI

3.13.1. Rozložení stránky

Grafické rozložení stránky je uvedeno na Obr. 18. Je realizováno pomocí HTML tagu div, který je blokovým elementem, což znamená, že zalamuje řádky před a po sobě. Každý obdélník jiné barvy reprezentuje jeden blokový element a celý tento komplet je vložen do hlavního blokového elementu. Poloha těchto elementů je řešena CSS styly. Hlavní element má nastavenu šířku na 90%, což ve skutečnosti znamená, že zobrazovaná část stránky je na 90% plochy prohlížeče. Tento element je pak zarovnán na střed. Tím vznikne stránka, která má po obou stranách dva bílé pruhy o velikosti 5%. Je u něho také definováno odsazení shora. Ostatní elementy jsou uvnitř tohoto elementu, proto se jejich velikost a umístění od hlavního elementu odvíjí. Element obsahující hlavičku má nastavenu šířku 100% a vlastnost overflow na hidden, což znamená, že část obrázku, která je delší než zmíněných 90% hlavního elementu, nebude zobrazena. Je tedy možné vytvořit dostatečně dlouhý obrázek, který bude pokrývat i největší rozlišení uživatele a zároveň při malém rozlišení uživatele bude zobrazena jen část obrázku a nebude vytékat ze stránky. Dalším elementem je element left, který má nastavenu šířku na 15% a minimální šířku na 150px. Vlastnost minimální šířky je použita, aby se text uvedený v menu nezalamoval a menu mělo pořád šířku 150px. Tato vlastnost není bohužel podporována prohlížečem Internet Explorer. Element left má dále nastavenu vlastnost float na left, což způsobí, že elementy za ním následující ho budou obtékat po pravé straně. Element middle, tedy středová část stránek, má nastavenu šířku 84% a vlastnost float na left, což umožní vznik bílého prázdného pruhu o velikosti 1% mezi levým a pravým elementem. Po těchto elementech následuje speciální element pro zrušení obtékání (je pojmenován cleaner). V elementu left jsou pak umístěny tři elementy o stejných vlastnostech reprezentující hlavní, vedlejší menu a informace.



Obr. 18 : Rozložení stránky

3.13.2. Menu

Menu se skládá z nadpisu a jednotlivých položek menu. Nadpis je element div, ve kterém je obrázek a text. Jednotlivé položky jsou realizovány pomocí seznamu ul li, v případě submenu ol li. Samotná položka je odkaz, který má nastavenu hodnotu display na block, což způsobí, že při najetí kurzoru na položku se obarví celý řádek a nejen pozadí textu položky. Změna barvy při najetí na položku je provedena v CSS pomocí dynamické pseudotřídy hover. Ikona příslušné položky je obrázek nastavený pomocí CSS jako pozadí odkazu a zarovnaný vpravo.

Obr. 19 : Ukázka menu stránky

3.13.3. Formuláře a výpisy záznamů

Barvy políček formulářů, styl písma a velikost písma je realizován pomocí CSS. V tabulce mohou být tři druhy políček(<td>). Je to políčko s obyčejným textem, políčko s nadpisem nebo políčko s kontrolním prvkem. Políčko s nadpisem má na pozadí obrázek o šířce 1px, který má nastavenu vlastnost background-repeat na hodnotu repeat, a proto je obrázek roztažen po celé šířce nadpisu. Díky tomu, že pro vytvoření grafiky je využita CSS technologie, stačí pozměnit jednu hodnotu v CSS souboru a změní se u všech odpovídajících elementů.

3.14. Adresářová struktura aplikace

Jednotlivé zdrojové kódy jsou umístěny do adresářů podle jejich účelu. Adresář includes obsahuje typicky zdrojové kódy, které se vkládají do jiných kódů. Většinou obsahují definice funkcí. V adresáři admin jsou pak zdrojové kódy určené pro část uživatele administrátora, naopak v adresáři users jsou kódy potřebné pro generování stránek obyčejného uživatele. V adresáři classe jsou definice tříd. Tyto třídy používám pro zpracování sms zpráv. Posledním adresářem se soubory potřebnými pro chod aplikace je adresář image. Zde jsou umístěny obrázky většinou ve formátu GIF, které slouží pro vytvoření grafiky aplikace. Tento adresář navíc obsahuje podadresář icon, kde jsou umístěny ikony, které jsou použity v hlavním menu a ve výpisech záznamů. V hlavním adresáři je pak spouštěcí soubor index.php, CSS soubory aplikace, soubor nápovědy help.php, ini soubor sloužící k nastavení aplikace, SQL skript pro vytvoření databáze a nakonec soubory s kódem pro zalogování uživatele. Poslední adresář je user_img, ten však slouží pro ukládání náčrtů uživatelů, jak je popsáno v kapitole 3.10.6. Nastavení.

4. Bezpečnost

4.1. Bezpečnost přihlašování

Přihlašování je realizováno pomocí globálních proměnných $_SESSION. Tyto proměnné jsou uloženy na straně serveru. Při přihlašování se alokuje paměť pro příslušné proměnné SESSION na straně serveru a na straně uživatele se do proměnných cookie uloží ID odkazující na zaalokovanou paměť. Při další akci uživatele se toto ID odešle spolu s požadavkem pomocí protokolu http a je tedy možné opět pracovat s proměnnými SESSION příslušejícími tomuto uživateli. Bezpečnostním rizikem je zde odcizení tohoto ID, jež je možné zjistit ze zmíněných proměnných cookie, které jsou volně k dispozici v souboru na disku uživatele. Toto zneužití je však možné pouze do chvíle, než se autentizovaný uživatel odloguje. Poté totiž aplikace smaže všechny proměnné SESSION a případnému útočníkovi aplikace ohlásí, že časová platnost stránky vypršela. Nabourání systému je z tohoto hlediska závislé jen na získání údajů z uživatelova počítače, což je ze strany aplikace neovlivnitelné.

4.2. Ukládání hesel

Heslo uživatele je do databáze ukládáno jako hash hesla s využitím tzv. solení. Solení znamená, že se k heslu přidá náhodně vytvořený řetězec a až poté se vše dohromady zahashuje. Tato metoda zajišťuje, že i když útočník získá zmíněný hash a dokáže nalézt jeho vzor (což je při použití vhodné hashovací funkce skoro nemožné), nedokáže rozpoznat heslo, jelikož je toto změněno pomocí solení. Jako hashovací funkce byla zvolena haval256.

4.3. Šifrovaná komunikace pomocí TCP protokolu

Šifrovat tuto komunikaci je velice důležité, protože v odesílaných datech se objevují čísla HW zařízení a hesla potřebná pro autentizaci komunikace. Pro šifrování je použit systém RSA, který je založen na jednosměrné funkci s padacími vrátky, které reprezentují privátní heslo. Pro zašifrování zprávy jsou zapotřebí dva veřejné klíče *m* a *e*, kde *m* je modulo a *e* je exponent. Pomocí těchto komponent se zpráva zašifruje, z čehož vyplývá, že zprávu může zašifrovat kdokoliv. Odšifrovat ji může pouze osoba, která zná privátní klíč. Tento klíč zná pouze tvůrce webové aplikace a tvůrce aplikace v C++, se kterým se komunikuje přes TCP. Heslo je uloženo v souboru na serveru, který je chráněn proti čtení a není stažitelný z veřejné datové sítě.

4.4. HTTPS

Do budoucna by bylo dobré provozovat tuto aplikaci pomocí protokolu https. Jeho funkce je stejná jako u protokolu http s tím rozdílem, že je přenos šifrován a to pomocí SSL nebo TSL. Toto zabezpečení pak zabraňuje odposlechům, packet-sniffingu a man-in-the-middle útokům. Https navíc komunikuje pomocí implicitního TCP portu 443 na rozdíl od http, který používá port 80. Pro zavedení tohoto protokolu je však třeba získat certifikát vydaný některou z certifikačních autorit, která zaručí, že se vlastník certifikátu nevydává za někoho jiného a poskytne mu tedy důvěryhodnost.

5. Instalace

Instalace aplikace spočívá ve vytvoření databáze, jejího naplnění prvním uživatelem a nahráním příslušné adresářové struktury obsahující zdrojové kódy skriptů a obrázků do adresáře, ze kterého server spouští jednotlivé aplikace. Typicky je to adresář htdocs. Dále je pak nutné správně nastavit hodnoty ini souboru, ve kterém jsou informace potřebné pro připojení k databázi a k serveru. K tomuto bude sloužit instalační skript install.php.

6. Testování

Systém byl otestován pomocí metody white-box, což znamená testování založené na znalosti kódu. Toto testování spočívalo v procházení všech možných cest programu alespoň jedenkrát, tedy provedení obou variant u všech podmínek a procházení cyklů tak, aby byl vyzkoušen každý jednotlivý krok. Další fází bylo testování funkčnosti formulářů a jejich odolnosti proti nesprávně zadané hodnotě, zpracování případné chyby a nabídnutí uživateli novému vyplnění dané položky. Bezpečnost systému byla testována klasickými typy útoků na nedokonalost systému, jako je například zadání URL adresy některého z formuláře bez zalogování nebo zadání URL adresy na část pro admina obyčejným uživatelem. Systém však není úplně bezpečný vzhledem k tomu, že není zajištěna šifrovaná komunikace mezi klientem a webovým serverem, což by vyřešilo zavedení protokolu https. Testováno bylo i grafické rozhraní nezávislou osobou, zda je ovládání aplikace dostatečně intuitivní a jednoduché. Jako poslední provedený test byl test součinnosti s ostatními prvky tohoto projektu, kde se testovala funkčnost odesílání příkazů přes GSM síť a následné přijetí informací od HW zařízení, komunikace se serverem přes kanál TCP a zobrazitelnost dat uložených serverem (aplikací v C++). Systém jako celek byl testován jak pod operačním systémem Linux, tak pod WindowsXP. V obou případech aplikace fungovala bez problémů.

7. Závěr

V rámci této bakalářské práce se podařila vytvořit funkční aplikace, která poskytuje správu nad uživateli a alarmy. Uživatel může spravovat události s alarmy spojenými a řídit alarmy pomocí příkazů zasílaných přes síť GSM. Aplikace podporuje dvě HW zařízení. První je určeno do obytných prostor a druhé do hospodářského stavení.

Byla provedena úvodní studie, která obsahuje katalog požadavků na funkčnost systému, kontextový diagram definující aktéry a uživatelské role, model jednání ukazující události, na které musí systém reagovat, výběr vhodného softwaru a analýzu trhu. Dalším krokem byl návrh aplikace, který obsahuje datový model a jeho popis, tedy návrh databáze, funkční model vysvětlující interakci mezi jednotlivými objekty, scénáře použití, dynamický model popisující stavy aplikace, návrh akceptačních testů, tedy testů, zda aplikace splnila zadání a nakonec specifikaci základních funkcí. Ve třetí části je popsána implementace tohoto návrhu. V rámci implementace se vytvořila databáze pomocí MySQL, která byla posléze naplněna testovacími daty. Dále byla napsána aplikace v jazyce PHP generující HTML stránky, u kterých byl vyvinut vhodný design s využitím obrázků ve formátu GIF a technologie CSS. Součástí této webové aplikace je i uživatelská příručka rozdělena na dvě části pro uživatele administrátora a obyčejného uživatele.

Tato aplikace byla vytvořena takovým způsobem, aby mohla být nadále rozšiřována a mohla tak podporovat další HW zařízení a zlepšovat uživatelský komfort.

Seznam použité literatury

[1] Jesus Castagnetto, Harish Rawat, Sascha Schumann, Chris Scollo, Deepak Veliath, Programujeme PHP profesionálně, Computer Press, 2001

[2] Stehen Spainhour, Robert Eckstein, Webmaster v kostce (pohotová referenční příručka), Computer press, 1999

[3] php dokumentace, http://www.php.net

[4] mySQL dokumentace, http://dev.mySQL.com/doc/

[5] předmět X36WWW, http://amun.felk.cvut.cz/x36www/index.HTML

[6] předmět X36SIN, http://service.felk.cvut.cz/courses/X36SIN/

[7] předmět X36RSF, http://service.felk.cvut.cz/courses/X36RSF/

[8] internetová příručka, http://www.jakpsatweb.cz/

[9] konkurenční projekt, http://www.hwgroup.cz/products/damocles/index_cz.html

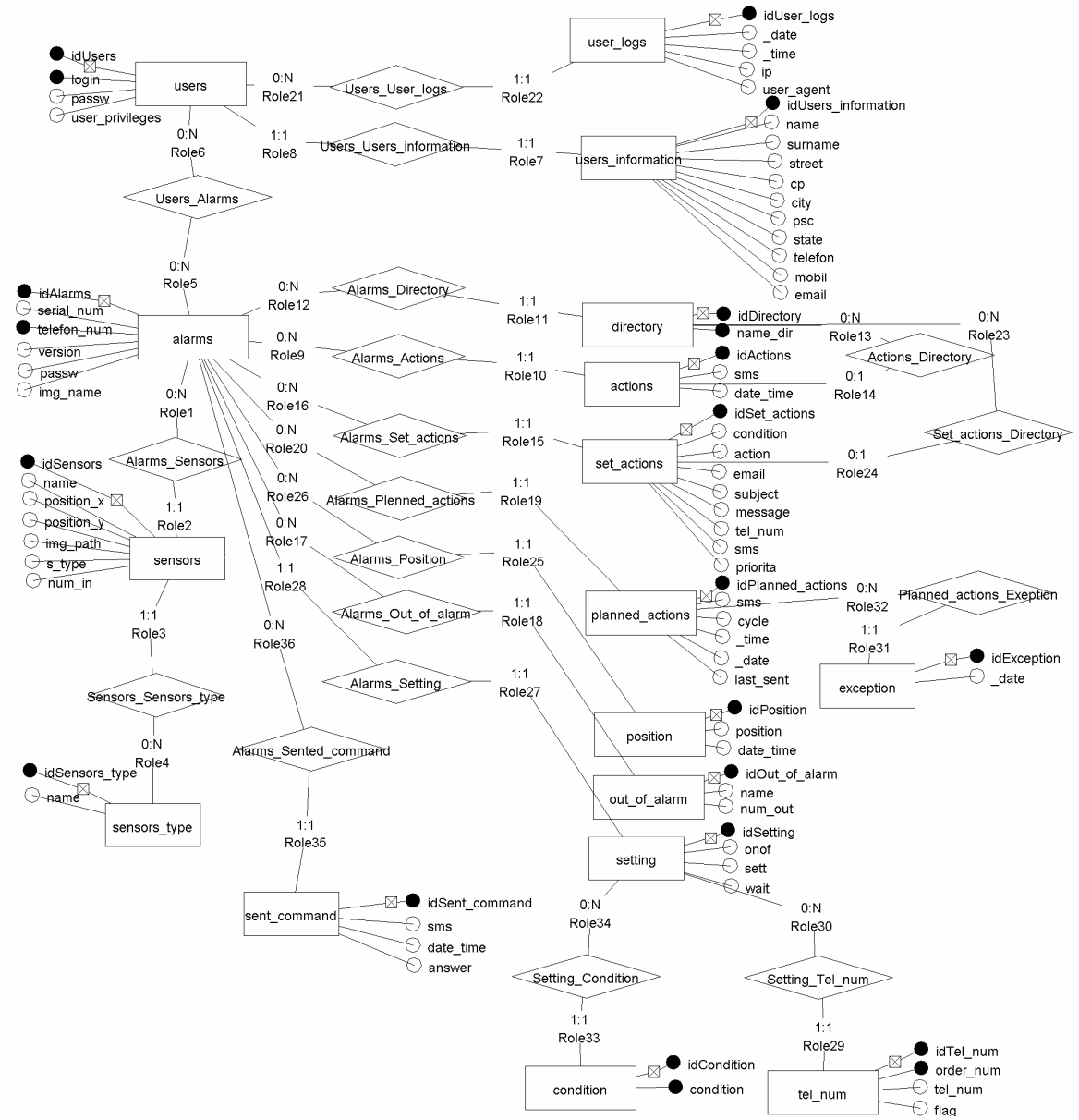
[10] zdroj pro zobrazení map, www.mapy.cz

Obsah přiloženého CD

Seznam příloh

1. Datový model (E-R diagram)
2. Podrobný popis datového modelu

Datový model (E-R diagram)



Podrobný popis datového modelu

Název entity
Popis entity.

Atribut 1	popis atributu 1
.	.
.	.
Atribut N	popis atributu N

users
Obsahuje údaje nutné pro zalogování uživatele.

idUsers	primární klíč
login	logovací jméno uživatele
passw	logovací heslo uživatele
users_privilages	práva uživatele

users_information
Obsahuje podrobné informace o uživateli.

idUsers_information	primární klíč
Users_idUsers	cizí klíč odkazující na entitní typ users
name	křestní jméno uživatele
surname	příjmení uživatele
street	adresa uživatele – ulice
cp	adresa uživatele – číslo popisné
city	adresa uživatele – město
psc	adresa uživatele – poštovní směrovací číslo
state	adresa uživatele – stát
telefon	telefon uživatele – pevná linka
mobil	telefon uživatele – mobilní telefon
email	emailová adresa uživatele

user_logs
Obsahuje informace o přihlášení uživatele do webové aplikace.

idUser_logs	primární klíč
user_id	cizí klíč odkazující na entitní typ users
_date	čas přihlášení uživatele
_time	datum přihlášení uživatele
ip	ip adresa uživatele
user_agent	webový prohlížeč uživatele

alarms
Informace o HW zařízení (alarmu).

idAlarms	primární klíč

seriove_c	sériové číslo alarmu
telefon_c	telefonní číslo telefonu připojeného k alarmu
verze	hardwarová/softwarová verze alarmu
passw	heslo alarmu, používané v sms příkazech pro bezpečnost komunikace
img_name	cesta ke schématu hlídaného objektu, tj. v případě budovy půdorys

sensore
Informace o čidlech připojených k danému alarmu.

idSensors	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
sensore_type_idSensors_type	cizí klíč odkazující na entitní typ senzore_type
name	název čidla / název umístění čidla
positron_x	pozice čidla ve schématu objektu, osa x
positron_y	pozice čidla ve schématu objektu, osa y
img_path	cesta k obrázku zobrazovaném na schématu objektu
type	typ čidla z hlediska jeho návratové hodnoty Dvě hodnoty: s – spínací čidlo (vrací 0 nebo 1) h – čidlo s číselnou návratovou hodnotou, např. teplotní čidlo vracející teplotu
Num_in	číslo portu připojení čidla na reálném přípravku

sensors_type
Možné typy čidel, např. pohybové čidlo.

idSensors_type	primární klíč
Name	jméno typu čidla

directory
„Složky“, do kterých se budou ukládat jednotlivé události, respektive se k události přiřadí cizí klíč složky.

idDirectory	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
name_dir	jméno složky

actions
Vzniklé události na alarmu v závislosti na nastavení alarmu. Například při sepnutí čidla alarm zašle sms serveru, který tuto událost uloží do tabulky actions.

idActions	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
sms	sms doručená od alarmu
date_time	čas doručení sms

set_actions
filtry aplikované na sms doručené od alarmu. Podporuji tři filtry a jejich vzájemné kombinace. Odeslání emailu, odeslání sms, přesunutí události do určené složky.

idSet_actions	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
direktory_idDirectory	cizí klíč odkazující na entitní typ directory
condition	podmínka pro aplikaci filtru na událost Tři hodnoty: 1 – vždy, 2 – při spuštění alarmu, 3 – reakce na plánovanou akci nebo vyžádanou sms
action	Akce, která se vykoná při splnění podmínky Tři hodnoty: 1 – přesun do složky, 10 – odeslání emailu, 100 – zaslání sms na uložené telefonní číslo, Možné hodnoty: 1,10,11,100,101,110,111
email	emailová adresa pro odeslání emailu
subjekt	předmět emailu
message	obsah emailu
tel_num	telefonní číslo, na které se sms přepošle
sms	sms, která se bude posílat
priorita	priorita filtru, filtr může mít prioritu 1, 2, 3. Priorita znamená důležitost vykonání filtru. Filtr s prioritou 1 je nejdůležitější a jeho zpracování je primárně nutné.

planned_actions
plánované akce pro odesílání příkazů alarmu

idPlanned_actions	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
sms	sms, která se odešle
cycle	parametr určující, zda je akce cyklická nebo ne. V případě, že ano, nastaví se rok, měsíc, týden, den, např. 2004040102 = rok 2004, měsíc duben, první týden,druhý den, tedy úterý. V případě, že akce není cyklická, nastaví se na 0.
_time	čas pro odeslání sms
_date	datum pro odeslání (v případě jednorázové akce)
last_sent	čas poslední odeslané akce

exception
Pokud uživatel nechce smazat celou cyklickou akci, ale jen v jeden den a hodinu udělat výjimku a tuto akci neprovést, uloží se tato výjimka do tabulky exception.

idException	primární klíč
planned_actions_idPlanned_actions	cizí klíč odkazující na entitní typ planned_actions
_date	datum výjimky

position
Slouží pro průběžné ukládání pozice objektu (využíváno hlavně pro automobil).

idPosition	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
position	zeměpisné souřadnice objektu.
date_time	čas uložení pozice

out_of_alarm
Výstupy alarmu. Například siréna.

idOut_of_alarm	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
name	jméno výstupu
num_out	číslo portu připojení výstupu na reálném přípravku

setting
Slouží k uložení nastavení HW přípravku.

idSetting	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
onof	alarm je zapnutý/vypnutý (0/1). Společné pro všechny alarmy.
wait	alarm zasílá/nezasílá (0/1) potvrzení o přijetí příkazu. Společné pro všechny alarmy.
sett	ve řetězci jsou uloženy příkazy nastavení alarmu oddělené čárkou. alarm home 1: TEMP<param>,OUTP<param>,INPU<param>,SPIN<param>

condition
Nastavení alarmu – podmínky pro aktivaci výstupu.

idCondition	primární klíč
setting_idSetting	cizí klíč odkazující na entitní typ setting
condition	Ve řetězci je uložen příkaz pro nastavení podmínky. alarm home 1: SETR<param>

tel_num
Nastavení alarmu – telefonní čísla uložená v alarmu, na které bude odesílat alarm sms v případě jeho aktivace nebo od kterých může přijímat příkazy pro nastavení.

idTel_num	primární klíč
setting_idSetting	cizí klíč odkazující na entitní typ setting
order_num	pozice čísla v tabulce čísel HW alarmu
tel_num	uložené telefonní číslo
flag	určuje typ telefonního čísla alarm home 1: s – čísla, ze kterých je možno zasílat příkazy pro nastavení r – čísla, na která se bude posílat informativní sms

sent_command
Slouží pro ukládání úspěšně odeslaných příkazů z webové aplikace.

idSent_command	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
sms	uložen pouze příkaz z odeslané sms
date_time	čas odeslání příkazu

users_has_alarm
Slouží pro ukládání úspěšně odeslaných příkazů z webové aplikace.

idUsers_has_alarm	primární klíč
Alarms_idAlarms	cizí klíč odkazující na entitní typ alarms
Users_idUsers	cizí klíč odkazující na entitní typ users