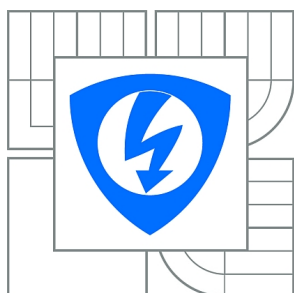


**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**  
BRNO UNIVERSITY OF TECHNOLOGY



**FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH  
TECHNOLOGIÍ**  
**ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY**

**FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION  
DEPARTMENT OF CONTROL AND INSTRUMENTATION**

## **INFORMAČNÍ SYSTÉM PRO MĚŘIČE ENERGIÍ**

**INFORMATION SYSTEM FOR ENERGY METERS**

**BAKALÁŘSKÁ PRÁCE**  
BACHELOR'S THESIS

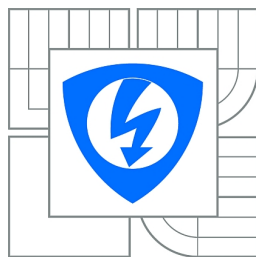
**AUTOR PRÁCE**  
AUTHOR

**FILIP MAGIC**

**VEDOUCÍ PRÁCE**  
SUPERVISOR

**Ing. LEŠEK FRANEK**

**BRNO 2015**



VYSOKÉ UČENÍ  
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky  
a komunikačních technologií

Ústav automatizace a měřicí techniky

# Bakalářská práce

bakalářský studijní obor  
**Automatizační a měřicí technika**

**Student:** Filip Magic

**ID:** 154796

**Ročník:** 3

**Akademický rok:** 2014/2015

**NÁZEV TÉMATU:**

## Informační systém pro měřiče energií

### POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je vytvoření informačního systému pro zobrazení spotřeby elektřiny, vody, plynu, tepla atd. za účelem rozúčtování, sledování bilancí a hledání optimalizací.

1. Zpracujte rešerši týkající se Smart Grid a Smart Meteringu.
2. Zpracujte analýzu případů užití systému.
3. Rozšiřte stávající datový model o prvky informačního systému.
4. Vytvořte SOAP server pro příjem dat z měřičů.
5. Vytvořte informační systém.
6. Otestujte funkčnost jednotlivých částí systému.

### DOPORUČENÁ LITERATURA:

ŘEPA, V. Analýza a návrh informačních systémů. Ekopress, 1999. ISBN 9788086119137. Dostupné z: <http://books.google.cz/books?id=gBr5AAAACAAJ>

BRUCKNER TOMÁŠ, V.J.B.A. Tvorba informačních systémů. Grada, 2012. ISBN 9788024741536. Dostupné z: <http://books.google.cz/books?id=9AAM7vnjK84C>

**Termín zadání:** 9.2.2015

**Termín odevzdání:** 25.5.2015

**Vedoucí práce:** Ing. Lešek Franek

**Konzultanti bakalářské práce:**

**doc. Ing. Václav Jirsík, CSc.**

*Předseda oborové rady*

### UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

## ABSTRAKT

Cieľom tejto bakalárskej práce je navrhnúť a vytvoriť informačný systém pre spracovanie dát z meračov energií.

Prvá časť práce je venovaná princípom inteligentných sietí. V ďalšej časti práce sú popísané webové technológie potrebné na vývoj SOAP servera a informačného systému.

Nasleduje návrh a vývoj SOAP serveru a informačného systému. V týchto kapitolách sú vytvorené diagramy prípadov použitia a návrh databázového modelu. Pri vývoji informačného systému bol použitý PHP framework Nette a databáza typu PostgreSQL.

## KLÚČOVÉ SLOVÁ

Nette, PHP, PostgreSQL, Bootstrap, Informačný systém, Merače energie, SOAP

## ABSTRACT

The aim of the Bachelor Thesis is to design and develop an information system for processing data from energy meters.

First part of the thesis is devoted to the principles of smart grids. In the next part are described web technologies necessary for the development of SOAP server and information system.

Furthermore is presented the proposal and development of the SOAP server. In these chapters are created use case diagrams and the data model. For the information system development was used PHP framework Nette and PostgreSQL database.

## KEYWORDS

Nette, PHP, PostgreSQL, Bootstrap, Information system, Energy meters, SOAP

## PREHLÁSENIE

Prehlasujem, že som svoju bakalársku prácu na tému „Informační systém pro měřiče energie“ vypracoval samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej bakalárskej práce ďalej prehlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/nebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona č. 121/2000 Sb., o právu autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), vo znení neskorších predpisov, vrátane možných trestnoprávných dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka č. 40/2009 Sb.

Brno .....

.....

(podpis autora)

## POĎAKOVANIE

Rád by som sa poďakoval vedúcemu bakalárskej práce pánovi Ing. Leškovi Fránekovi za odborné vedenie, konzultácie, nekonečnú trpezlivosť a podnetné návrhy k práci.

Brno .....

.....

(podpis autora)

# OBSAH

<b>Úvod</b>	<b>11</b>
<b>1 Inteligentné siete (Smart Grids)</b>	<b>12</b>
1.1 Rozvoj inteligentných sietí vo svete . . . . .	13
1.1.1 Česko . . . . .	13
1.1.2 Čína . . . . .	13
1.1.3 USA . . . . .	14
1.1.4 Japonsko . . . . .	14
1.1.5 Nemecko . . . . .	14
1.2 Komponenty v inteligentných sietach . . . . .	14
1.2.1 Vodomery, plynomery, atď. . . . .	14
1.2.2 LCD panely, mobilné telefóny, televízie . . . . .	14
1.2.3 Elektromery . . . . .	15
1.2.4 Datový koncentrátor . . . . .	15
1.2.5 Server . . . . .	16
<b>2 Použité technológie</b>	<b>17</b>
2.1 Protokoly a programovacie jazyky . . . . .	17
2.1.1 HTTP . . . . .	17
2.1.2 HTML . . . . .	17
2.1.3 CSS . . . . .	18
2.1.4 PHP . . . . .	18
2.1.5 JavaScript . . . . .	19
2.1.6 XML . . . . .	19
2.1.7 WSDL . . . . .	20
2.1.8 SOAP . . . . .	20
2.1.9 Webové služby . . . . .	21
2.2 Databázový systém . . . . .	22
2.2.1 PostgreSQL . . . . .	22
2.3 Knižnice a frameworky . . . . .	22
2.3.1 Bootstrap . . . . .	22
2.3.2 JQuery . . . . .	23
2.3.3 morris.js . . . . .	23
2.3.4 Dibi . . . . .	23
2.4 Nette Framework . . . . .	23
2.4.1 MVC prístup . . . . .	23
2.4.2 Zabezpečenie . . . . .	24

<b>3</b>	<b>SOAP Server</b>	<b>26</b>
3.1	Analýza dát a metód . . . . .	26
3.1.1	Metóda DataService . . . . .	26
3.1.2	Dátový typ dataRequest . . . . .	27
3.1.3	Dátový typ dataResponse . . . . .	29
3.2	Databázový model pre namerané dáta . . . . .	30
3.2.1	Identifikácia entít . . . . .	30
3.2.2	ER diagram . . . . .	32
3.3	Vývoj SOAP server . . . . .	33
3.3.1	Použité technológie . . . . .	33
3.3.2	Adresárová štruktúra . . . . .	33
3.3.3	Realizácia . . . . .	33
3.3.4	Testovanie . . . . .	36
<b>4</b>	<b>Návrh informačného systému</b>	<b>37</b>
4.1	Analýza požiadaviek . . . . .	37
4.1.1	Všeobecné požiadavky . . . . .	37
4.1.2	Rozšírené funkčné požiadavky . . . . .	37
4.1.3	Diagramy prípadov použitia . . . . .	38
4.2	Návrh databázového modelu pre informačný systém . . . . .	41
4.2.1	Identifikácia entít . . . . .	41
4.2.2	ER diagram . . . . .	41
4.3	Adresárová štruktúra systému . . . . .	43
4.4	Architektúra systému . . . . .	43
4.4.1	Model . . . . .	43
4.4.2	Presentery . . . . .	44
4.4.3	Šablóny . . . . .	45
4.5	Realizácia . . . . .	46
4.5.1	Zabezpečenie proti nepovolenému vstupu . . . . .	46
4.5.2	Formuláre . . . . .	47
4.5.3	Tabuľkový výpis dát . . . . .	48
4.5.4	Grafické zobrazenie nameraných hodnôt . . . . .	48
4.5.5	Ukážky užívateľského rozhrania . . . . .	49
<b>5</b>	<b>Záver</b>	<b>53</b>
	<b>Literatúra</b>	<b>54</b>
	<b>Zoznam symbolov, veličín a skratiek</b>	<b>56</b>

Zoznam príloh	57
A Obsah priloženého CD	58



# ZOZNAM OBRÁZKOV

1.1	Cieľový stav inteligentých sietí [4]	12
1.2	Dátový koncentrátor MT34A	15
2.1	Schéma komunikácie webovej aplikácie	19
2.2	Webové služby	21
3.1	Komunikácia medzi SOAP serverom a klientom	26
3.2	Dátový typ dataRequest	28
3.3	Dátový typ dateResponse	29
3.4	ER diagram pre namerané dáta	32
3.5	UML model triedy DataClass	34
3.6	Simulátor odosielania nameraných dát pomocou protokolu SOAP	36
4.1	Diagram prípadov použitia - Neprihlásený užívateľ	38
4.2	Diagram prípadov použitia - Zákazník	39
4.3	Diagram prípadov použitia - Dodávateľ	40
4.4	Diagram prípadov použitia - Administrátor	40
4.5	ER diagram informačného systému	42
4.6	Prihlásenie do systému	49
4.7	Úvodná obrazovka zobrazená prihlásenému užívateľovi	50
4.8	Administrácia meračov - Zoznam meračov	50
4.9	Detailné zobrazenie konkrétneho merača	51
4.10	Editácia oprávnení prístupu k meračom	52

## ZOZNAM TABULIEK

3.1	Popis premenných v dátovom type dataRequest . . . . .	29
3.2	Popis premenných v dátovom type dataRequest . . . . .	30
3.3	Popis entít v databázovom modeli pre namerané dáta . . . . .	31
3.4	Popis entít v databázovom modeli pre driver . . . . .	31
3.5	Súborová štruktúra SOAP server . . . . .	33
4.1	Popis entít v databázovom modeli pre informačný systém . . . . .	41
4.2	Zoznam väzobných tabuliek . . . . .	41
4.3	Zoznam šablón . . . . .	46

# ÚVOD

Cielom práce je navrhnuť a vytvoriť informačný systém pre merače energií. Hlavné uplatnenie by mal systém nájsť v oblasti inteligentných sietí a inteligentného merania spotreby. Úlohou systému je spracovanie dát nameraných na meračoch energie, napríklad meračoch spotreby elektrickej energie, vody, plynu, tepla atď. Informačný systém tieto dáta prezentuje užívateľom, ktorý ich môžu využiť na presnejšie sledovanie svojej spotreby a dodávateľom energií poskytuje podklady pre rozúčtovanie. Typické aplikácie systému sú napríklad študentské internáty, obchodné centrá alebo bytové družstvá.

Súčasťou práce je návrh a tvorba SOAP servera, ktorého úlohou je spracovať dáta posielať dátovým koncentrátorom a následne spracované dáta poskytnúť informačnému systému. Práce je realizovaná v spolupráci so spoločnosťou Modemtec s.r.o, ktorá poskytla popis komunikácie medzi dátovým koncentrátorom a SOAP serverom. K dispozícii je tiež simulátor dátového koncentrátoru, ktorý odosiela simulované dáta pomocou protokolu SOAP v reálnom čase.

K úspešnému vyriešeniu zadaných cieľov bude nutné zoznámiť sa s princípmi inteligentných sietí, komunikačnými protokolmi a s vývojom informačných systémov. Na základe získaných poznatkov bude následne zvolený vhodný návrh a technológie pre realizáciu samotného informačného systému.

# 1 INTELIGENTNÉ SIETE (SMART GRIDS)

Intelligentné siete (Smart Grids) v sebe kombinujú tradičné technológie s inovatívnymi digitálnymi riešeniami, čím rastie pružnosť riadenia elektrickej siete vďaka efektívnejšej výmene informácií. Jedným z posledných uplatnení inteligentných sietí je sieťová integrácia jednotlivých prevádzok výroby elektriny z obnoviteľných zdrojov, ktorých rozmach sleduje naplnenie environmentálnych cieľov stanovených Európskou komisiou. Použitie inovatívnych digitálnych technológií umožňuje monitorovať celú sieť a regulovať energetické toky ako prevenciu pred výpadkami, čím je možné dosiahnuť bezchybné zásobovanie energiami.

V tomto nepretržite sa vyvíjajúcom systéme sa zo zákazníkov stávajú aktívni účastníci energetickej sústavy s transparentnými tokmi energie, ktorí vstupujú na trh s energiou a efektívne ju využívajú. [1] Cieľový stav integrácie inteligentných sietí do systému prenosov energií môžete vidieť na obrázku č. 1.1



Obr. 1.1: Cieľový stav inteligentných sietí [4]

Objekty označené číslami na obrázku č. 1.1 majú nasledujúci význam[4]:

1. Veľkokapacitné elektrárne
2. Alternatívne zdroje energie (veterné farmy, solárne panely) - elektrina z obnoviteľných zdrojov je používaná na vyrovnanie dopytu a ponuky v sieti
3. Kogeneračná jednotka umiestnená v mieste spotreby
4. Elektromobil, vrátane infraštruktúry verejných nabíjacích staníc
5. Automatizované kontrolné centrum
6. Smart Meters, merače umožňujúce obojsmerný prenos informácií po sieti
7. Elektromobil slúžiaci ako akumulátor, vyrovnáva energiu v sieti odčerpávaním uskladnenej energie z pripojených batérií
8. Skladovanie energie počas času nižšej spotreby v batériách, energia je neskôr znova použitá v špičke
9. Diaľkové ovládače a senzory
10. Izolovaná časť distribučnej siete

Vďaka inteligentným sieťam budú mať odberatelia energií lepší prehľad o svojej spotrebe v reálnom čase. Zákazníkom sa tak otvára možnosť svoju spotrebu viacej ovplyvňovať. Použitie inteligentných meračov umožní prepínanie typu tarífov podľa vyťaženia siete, alebo zmluvných podmienok, čo ponúka nové možnosti pre zákazníkov. Namerané dáta zase umožnia optimalizovať spotrebu a výrobu elektrickej energie, čím sa znížia celkové náklady. [5]

## 1.1 Rozvoj inteligentných sietí vo svete

### 1.1.1 Česko

Spoločnosť ČEZ v roku 2010 spustila pilotný projekt zameraný na testovanie inteligentných sietí v lokalitách Pardubice, Jeřmanice a Vrchlabí. V pláne je nainštalovať zhruba 40 tisíc inteligentných elektromerov, ktoré majú priniesť užívateľom nové možnosti v rozhodovaní o využití energií. [5]

### 1.1.2 Čína

Čínska vláda poskytla značnú sumu financií na podporu rozvoja využitia inteligentných sietí pri správe vodných systémom, cestnej infraštruktúry a elektrických sietí. Energetická spoločnosť State Grid Corporation of China v roku 2010 schválila program na podporu rozvoja inteligentných sietí trvajúci do roku 2030. Výška investície by do roku 2020 mala dosiahnuť 96 miliárd dolárov. [2]

### **1.1.3 USA**

V roku 2009 bola zákonom American Recovery Reinvestment Act vyčlenená suma 4.5 miliardy dolárov na podporu rozvoja sietí. Táto suma bola určená na pokrytie nákladov potrebných na integráciu inteligentných sietí do existujúcej energetickej siete. Časť financií, konkrétne 425 miliónov dolárov, bola použitá na prezentáciu výhod smart grids a skladovania energií.[2]

### **1.1.4 Japonsko**

Federácia FEPC združujúca energetické spoločnosti v Japonsku vyvíja inteligentnú sieť, ktorej cieľom je zlepšiť využitie energií generovaných obnoviteľnými zdrojmi. Projekt je z časti financovaný Japonskou vládou, ktorá taktiež stanovila iniciatívu na využitie nových technológií v národných prenosových sústavách. [2]

### **1.1.5 Nemecko**

Konzorcium firiem v roku 2009 vytvorilo pilotný projekt MeRegio na zavedenie inteligentných sietí v priemyselnom regióne Karlsruhe-Stuttgart nachádzajúcom sa v Južnom Nemecku. V projekte je zapojených zhruba 1000 odberateľov z domácností, priemyselných podnikov, výrobných a skladovacích jednotiek. V Nemecku sa realizujú aj iné pilotné projekty, napríklad MoMa v Mannheime a veľký prevádzkari distribučných sietí spúšťajú vlastné testovania Smart Grids. Nemecká vláda výrazne podporuje projekty zamerané na koncept inteligentných sietí. [5]

## **1.2 Komponenty v inteligentných sieťach**

### **1.2.1 Vodomery, plynomery, atď.**

V tejto kategórii sa nachádzajú zariadenia, ktoré slúžia na meranie rôznych fyzikálnych veličín s výnimkou elektrickej energie. Okrem uvedených sem patria napríklad kalorimetry, teplomery alebo hladinomery. Tieto zariadenia komunikujú s elektromerom pomocou komunikačného rozhrania a predávajú mu namerané údaje. Komunikáciu zabezpečujú technológie M-Bus, Wireless M-Bus, ZigBee, poprípade impulzy v prípade použitia impulzných meradiel.[3]

### **1.2.2 LCD panely, mobilné telefóny, televízie**

Účelom týchto zariadení je poskytnúť užívateľom informácie o spotrebe energií, aktivovanom účtovacom tarife a ľubovoľné dodatočné informácie. Zákazníkovi sú poskyt-

nuté informácie o aktuálnej účtovacej tarife, čo mu umožňuje meniť svoju spotrebu a tým zabezpečiť úsporu energií a nákladov. [3]

### 1.2.3 Elektromery

Primárnym účelom elektromerov je meranie spotreby elektrickej energie, monitorovanie napätia v inštalačnom mieste a zaznamenávanie iných užitočných udalostí týkajúcich sa danej prípojky. V závislosti na inteligencii merača môže elektromer meniť nastavený typ tarifu, limitovať odber alebo odpojiť zákazníka. Namerané dáta z elektromeru sú následne predávané do dátového koncentrátora. Prenos nameraných údajov je možné zabezpečiť pomocou modemu pre PLC sieť, ktorý môže byť súčasťou elektromeru alebo je pripojený modulárne. Ak nie je možné použiť PLC komunikáciu je nahradená službou GPRS, rádiovým prenosom poprípade v priemyselných aplikáciách metalickým vedením ako je M-BUS, RS-485 alebo Ethernet. [3]

### 1.2.4 Dátový koncentrátor

Úlohou dátového koncentrátora je tvorba rozhrania, slúžiaceho na prenos údajov, medzi elektrickou, poprípade rádiovou sieťou a inými typmi komunikácie, najčastejšie TCP/IP. Na dátový koncentrátor môže byť pripojených zhruba 100 zariadení, ale toto číslo môže rásť až na 1000 v priemyselných aplikáciách v závislosti na type dátového koncentrátora. Namerané údaje sú po ich spracovaní odoslané na server pomocou technológií LAN, WIFI, alebo GPRS spojenia.[3]



Obr. 1.2: Dátový koncentrátor MT34A

### 1.2.5 Server

Server slúži na spracovávanie dát zasielaných dátovými koncentrátormi. Časť týchto dát je následne poskytnutá klientským staniciam, ktoré tak majú k dispozícii aktuálne informácie o stave siete. Operátorom stanice je umožnené komunikovať s meračmi a meniť ich nastavenia, čo umožňuje ovládať merače, meniť typ tarifu podľa vyťaženia siete a zabrániť kolapsom siete. [3]



## 2 POUŽITÉ TECHNOLOGIE

### 2.1 Protokoly a programovacie jazyky

#### 2.1.1 HTTP

HTTP je protokol, ktorý definuje ako sú informácie internetom prenášané a formátované. Tento protokol primárne slúžil na prenos hypertextových súborov, ale aktuálne verzie zvládajú prenášať všetky druhy dát s pomocou MIME (Multipurpose Internet Mail Extensions).

HTTP je bezstavový protokol, ktorý rieši komunikáciu medzi klientom a serverom, pričom komunikácia je založená na princípe odoslania požiadavky klientom - odoslanie odpovede serverom. Každý príkaz je na cieľovom servery spracovávaný samostatne a nemá povedomie o predchádzajúcich príkazoch, ktoré boli od daného klienta odoslané. To spôsobuje problémy pri identifikácii klientov a inteligentných reakciách webových aplikácií na užívateľské vstupy. Tento problém sa rieši využitím ďalších technológií ako sú Cookies, Javascript alebo ActiveX.[6]

Príkladom zavolania http protokolu je zadanie URL adresy do webového prehliadača a jej potvrdenie. Tento krok odošle na server http požiadavok, ktorý je spracovaný serverom a odošle ako odpoveď dáta o požadovanej webovej stránke.

#### 2.1.2 HTML

HTML je značkovací jazyk zameraný na vytváranie webových stránok. Prvá verzia jazyka HTML sa datuje do roku 1993, keď ju vytvoril Tim Berners-Lee, ako podmnožinu jazyka SGML. V súčasnosti je HTML spravované medzinárodným konzorciom World Wide Web Consortium(W3C). Súčasná stabilná verzia HTML je HTML 5.0 publikovaná 28. 10.2014 a momentálne je vo vývoji verzia 5.1.

Jazyk HTML sa skladá z krátkych značiek, ktoré slúžia na formátovania dokumentu a jeho obsahu. Dokument je následne zobrazený webovým prehliadačom, ktorý HTML spracuje a prezentuje v grafickej podobe. HTML obsahuje niekoľko značiek, ich názvy sú v dokumente zapisované medzi uhlové zátvorky('<' a '>'). Časť dokumentu obsahujúca otváraciu značku, uzatváraciu značku a obsah ohraňovaný týmito značkami tvorí element dokumentu a z elementov je tvorený celý dokument.[7]

Základné značky používané v HTML:

- <!DOCTYPE html> - značka definujúca verziu štandardu, ktorý je použitý
- <html> - informuje prehliadač o tom, že je použitý jazyk html

- <head> - element v ktorom sú definované informácie o dokumente
- <title> - názov dokumentu
- <body> - element popisujúci obsah dokumentu

Výpis kódu 2.1: Ukážka formátu HTML dokumentu

```

1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>Názov dokumentu...</title>
5    </head>
6    <body>
7      Obsah dokumentu..
8    </body>
9  </html>

```

### 2.1.3 CSS

CSS je jazyk slúžiaci na popísanie formátovania a zobrazovania informácií na dokumentoch vytvorených s pomocou HTML, XHTML alebo XML. Definuje štýl jednotlivých elementov webovej stránky, umožňuje nastaviť farby, typ písma, rozloženie stránky atď.[8]

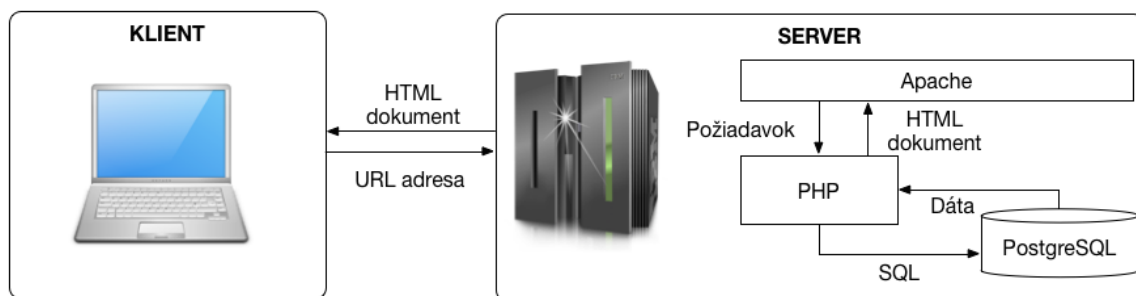
Primárnou výhodou CSS je možnosť oddeliť vzhľad webovej stránky od jej obsahu, čo umožňuje znížiť zložitosť a opakovanie zdrojového kódu.

### 2.1.4 PHP

PHP je skriptovací programovací jazyk vytvorený v roku 1994 Rasmusom Lerdofom. Jeho cieľom bolo vytvoriť programovací jazyk na tvorbu jednoduchých dynamických webových aplikácií. Svoje dielo vydal prvý krát v roku 1994 pod názvom Personal Home Page Tools, skrátene PHP Tools.

V súčasnosti patrí PHP medzi najpopulárnejšie skriptovacie jazyky a je vo veľkej miere využívaný pri vývoji. Za svoju popularitu vďačí možnosti integrovať php kód priamo do html kódu a tým prepojiť dynamickú a statickú časť dokumentu.

Jazyk PHP v sebe zahrňuje množstvo knižníc slúžiacich na spracovávanie textov, súborov, generovanie grafiky, ovládače na pripojenie k najpopulárnejším databázovým serverom ( MySQL, PostgreSQL, Oracle...) a má v sebe implementovanú podporu pre najpoužívanéjšie protokoly (HTML,FTP,HTTPS,...).[9]



Obr. 2.1: Schéma komunikácie webovej aplikácie

### 2.1.5 JavaScript

Jedná sa o skriptovací programovací jazyk, ktorý je spracovávaný na strane klienta. Používa sa na pridanie dynamických prvkov a skvalitnenie webového rozhrania pre klientov. Umožňuje zmeniť obsah webovej stránky bez toho, aby užívateľ bol nútený ručne vykonať obnovenie obsahu. V praxi sa využíva na animácie, automatickú kontrolu formulárov pred ich odoslaním, zobrazovanie atď.

Skript napísaný v tomto jazyku je vo forme textu vloženom do dokumentu v jazyku HTML a pri návšteve webovej stránky je spustený. Nevýhoda je nutnosť podpory jazyka Javascript zo strany webového prehliadača, ktorý nemusí podporovať aktuálnu verziu, alebo mať podporu vypnutú. Preto je výhodnejšie Javascript používať ako nadstavbu pre užívateľské prostredie, ale nevyžadovať ho pre využitie základnej funkcionality informačného systému.[10]

### 2.1.6 XML

XML celým názvom Extensible Markup Language, slovensky rozširiteľný značkovací jazyk, je obecný značkovací jazyk, ktorý definuje zoznam pravidiel na kódovanie dokumentov vo formáte, ktorý je čitateľný pre ľudí aj počítače.

Jazyk XML vychádza z SGML a bol vytvorený za účelom publikovania veľkého množstva dát v elektronickom formáte v súčasnosti je využívaný aj ako štandardný formát pri výmene informácií medzi aplikáciami.

Jazyk XML pracuje so značkami, ktoré ale nie sú narozdiel od jazyka HTML vopred definované a ich názvy sú vytvorené autorom XML dokumentu. Obsah ohraňovaný otváracou značkou a uzatváracou značkou spolu so značkami tvorí element XML dokumentu.[11]

Výpis kódu 2.2: Ukážka formátu XML dokumentu

```
1 | <list>
```

```

2   <autor>Peter</autor>
3   <obsah>Toto je obsah</obsah>
4 </list>

```

### 2.1.7 WSDL

WSDL je jazyk, ktorý popisuje funkcionality webových služieb a poskytuje definíciu názvu operácií, používaných dátových typov, komunikačného protokolu a popis vstupných a výstupných parametrov.

Dokumenty v jazyku WSDL sú popisované vo formáte XML, ktorý je jednoduchý na spracovanie počítačom a relatívne prehľadný aj pre človeka. Vo väčšine prípadov je WSDL formát používaný na popis prenosu pomocou protokolu SOAP. WSDL formát popisuje štruktúru SOAP servera a klienti podľa popisu poskytovanom v WSDL dokumente môžu zistiť aké služby sú dostupné na strane servera a ako ich správne volať.[12]

Výpis kódu 2.3: Jednoduchý príklad WSDL popisu:

```

1 <message name="getTermRequest">
2   <part name="term" type="xs:string"/>
3 </message>
4
5 <message name="getTermResponse">
6   <part name="value" type="xs:string"/>
7 </message>
8
9 <portType name="glossaryTerms">
10  <operation name="getTerm">
11    <input message="getTermRequest"/>
12    <output message="getTermResponse"/>
13  </operation>
14 </portType>

```

### 2.1.8 SOAP

SOAP je protokol vytvorený na výmenu dát vo formáte XML. Komunikácia nie je viazaná na žiadny konkrétny transportný protokol, ale väčšina služieb využíva HTML alebo SMTP protokol na prenos dát.

V SOAP sú definované pravidlá na odosielanie štrukturovaných správ, ktoré sa využívajú pri jednoduchej jednosmernej komunikácii, avšak v praxi sa využívajú hlavne pri RPC(vzdialenom volaní procedúr).

SOAP protokol nie je viazaný na konkrétny programovací jazyk a umožňuje jednoduché prepojenie aplikácií a tvorbu distribuovaných systémov. Najčastejšie sa využíva s webovými službami.[13]

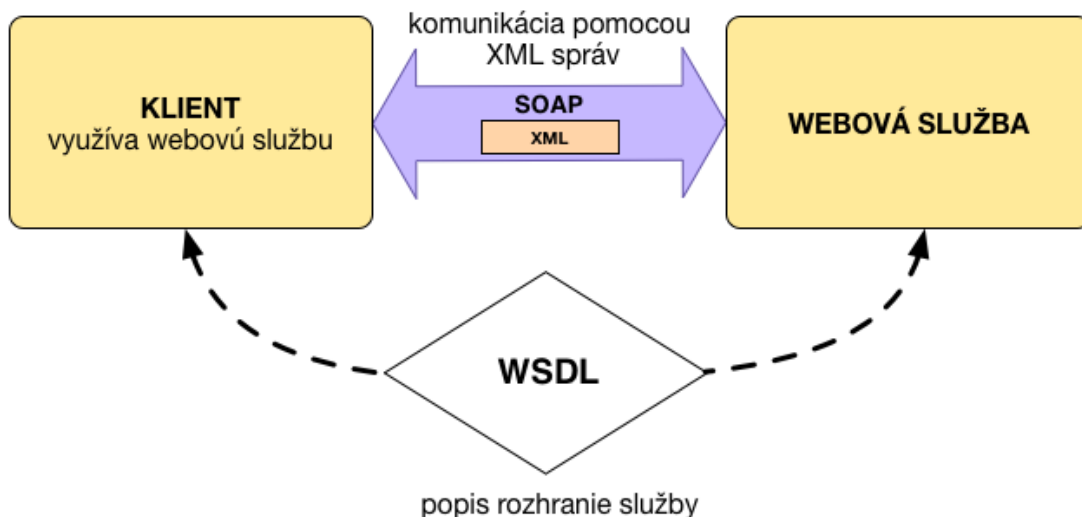
SOAP správa obsahuje:

- Deklarácia XML(voliteľné) - špecifikuje štandard XML využitý pri tvorbe SOAP správy
- SOAP obálka - obsahuje definíciu SOAP prenosu, špecifikácie a obsah správy
  - SOAP Hlavička (voliteľné) - obsahujú údaje slúžiace na autentifikáciu alebo správu session
  - SOAP Telo - popisuje prenášané údaje a definované metódy.

### 2.1.9 Webové služby

Termín webové služby popisujú štandardizovaný spôsob komunikácie medzi dvoma zariadeniami cez internetovú sieť. Využívajú sa pri tom XML, SOAP a WSDL. XML slúži na popis prenášaných dát, SOAP protokol zabezpečuje prenos dát medzi zariadeniami a WSDL slúži na popis služieb, ktoré sú zariadeniami poskytované.

Podobne ako SOAP nie sú webové služby viazané na konkrétny programovací jazyk, sú využívané pri tvorbe distribuovaných systémov a na prenos dát medzi aplikáciami.



Obr. 2.2: Webové služby

## 2.2 Databázový systém

### 2.2.1 PostgreSQL

Jedná sa o objektovo-relačný databázový systém (ORDBMS) vyvíjaný od roku 1982 na základe databázového systému Ingres. Tento systém je vydávaný pod open-source licenciou a na vývoji sa podieľa svetová komunita vývojárov a firiem .

Jeho primárnym účelom je ukladať bezpečne dáta a umožniť ich získavanie po zadaní požiadavky. PostgreSQL je vyvíjaný multiplatformovo, primárne na unix-like systémy a operačný systém Windows.

Databázový systém PostgreSQL je založený na štandarde SQL a tento jazyk je používaný na operáciu s dátami uloženými v databáze. PostgreSQL podporuje použitie triggerrov, sequencerov a je navrhnutý na prácu s veľkými datovými sadami pri zachovaní výkonu. [14]

Tento databázový systém bol zvolený, pretože sa jedná o firemný štandard používaný v spoločnosti Modemtec.

## 2.3 Knižnice a frameworky

Framework je knižnica naprogramovaných tried a objektov, ktorej cieľom je zjednodušiť vývoj aplikácií. Framework programátorovi poskytuje riešenia typických problémov a umožňuje mu sústrediť sa len na svoje zadanie. Použitie frameworku má za následok mierne zhoršenie výkonu aplikácie a väčšie nároky na server, avšak doba vývoja aplikácie bude značne skrátená.

### 2.3.1 Bootstrap

Bootstrap je front-endový framework vyvíjaný spoločnosťou Twitter od roku 2011. Je šírený pod licenciou MIT Licence a jedná sa o projekt s otvoreným zdrojovým kódom.

Cieľom tohto frameworku je zjednodušiť vývoj webových stránok, umožňuje rýchlym spôsobom navrhnuť webovú stránku založenú na responzívnom dizajne.

Bootstrap poskytuje zbierku nástrojov určených na vytváranie prostredia slúžiacu na komunikáciu medzi klientom a serverovou časťou aplikácie. Obsahuje rozsiahlu zbierku HTML a CSS šablón pre vytváranie typografie, foriem, tlačidiel, navigácie a iné webové elementy.[15]

### 2.3.2 JQuery

jQuery je javascriptová knižnica, vydávaná pod licenciou MIT a vyvíjaná JQuery Team od roku 2006. Cieľom jQuery je zjednodušenie práce s dokumentom(DOM), obsahuje funkcie ktoré pomáhajú používať animácie, ošetrovať udalosti a interakciu s Ajaxom. Výhodou jQuery je jeho jednoduchá rozširiteľnosť a možnosť vytvárať nové pluginy. Väčšina webových nástrojov ho v súčasnosti využíva, medzi ne patrí aj Bootstrap.[16]

### 2.3.3 morris.js

morris.js je javascriptová knižnica vydávaná pod licenciou BSD. Je pripravená na jednoduché generovanie grafov pomocou javascriptu a dát zadávaných vo formáte JSON. Je využívaná na zobrazenie nameraných hodnôt z meračov v informačnom systéme.

### 2.3.4 Dibi

Dibi je PHP databázový layer, ktorého cieľom je zjednodušiť vytváranie SQL príkazov a spríjemniť rutinné činnosti. Dibi obsahuje automatickú podporu konvencií ako je escapovanie a slashovanie. Ďalšou výhodou je automatické formátovanie špeciálnych typov ako sú dátumy alebo textové refazce.

Dibi sa riadi filozofiou maximálnej jednoduchosti a značne zlepšuje prehľadnosť SQL príkazov. Je ju možné využiť s databázami typu MySQL, PostgreSQL, SQLite, MS SQL, Oracle, Access, PDO a ODBC. Je veľmi jednoduché ju pripojiť k rôznym frameworkom a ja ju budem využívať v spolupráci s Nette.[17]

## 2.4 Nette Framework

Nette framework je český PHP framework, ktorý sa teší veľkej popularite medzi vývojármi. Jeho výhodou je široká podpora webových technológií a veľký doraz kladený na zabezpečenie. Dôvodov pre voľbu vývoja v Nette bolo viacej, okrem odporúčania tohto frameworku zo strany vedúceho práce, sa jednalo o kvalitne spracovanú dokumentáciu, aktívnu komunitu a veľké množstvo doplnkov a komponentov.

### 2.4.1 MVC prístup

Model-View-Controller je softvérová architektúra, ktorá vznikla z potreby oddeliť u aplikácií s grafickým rozhraním kód obsluhy (controller) od kódu aplikačnej logiky (model) a od kódu zobrazujúceho dáta (view). Tým sa aplikácia spríjemňuje,

uľahčuje budúci vývoj a umožňuje testovanie jednotlivých častí zvlášť.

## **Model**

Model je dátový a najmä funkčný základ celej aplikácie. Je v ňom obsiahnutá aplikčná logika. Akákoľvek akcia užívateľa (prihlásenie, vloženie tovaru do košíka, zmena hodnoty v databáze ) predstavuje akciu modelu. Model si spravuje svoj vnútorný stav a ako výstup ponúka len pevne dané rozhranie. Volaním funkcií tohto rozhrania môžeme zisťovať či meniť jeho stav. Model o existencii view alebo kontroľeru nevie. [18]

## **View**

View, voľne preložené pohľad, je vrstva aplikácie, ktorá má na starosti zobrazenie výsledku požiadavky. Obvykle používa šablonovací systém a ovláda ako sa má zobrazíť konkrétny komponent alebo modul. [18]

## **Controller**

Controller, ktorý spracováva požiadavky užívateľov a na ich základe volá patričnú aplikčnú logiku(Model) a následne požiada view o vykreslenie dát. Obdobu kontroľerov v Nette Framework-u predstavuje Presenter. [18]

### **2.4.2 Zabezpečenie**

#### **Cross-Site Scripting (XSS)**

Cross - Site Scripting je metóda narušenia webových stránok zneužívajúca neošetrené výstupy. Útočník dokáže do stránky podstrčiť svoj vlastný kód a tým môže stránku pozmeniť alebo dokonca získať citlivé údaje o návštevníkoch. Proti XSS sa možno brániť len dôsledným a korektným ošetrovaním všetkých reťazcov.

Nette Framework využíva technológiu Context-Aware Escaping, ktorá všetky výstupy ošetrí automaticky a tým znižuje možnosť vzniku bezpečnostnej chyby z dôvodu pozabudnutia, alebo nepozornosti. Príkladom takéto ošetrovania je napríklad výpis premennej `$message = 'Šírka 1/2'`, ktorá sa pri generovaní HTML kódu automaticky ošetrí v závislosti na jej použití. Napríklad v atribúte `onclick` by sa premenná vypísala ako `&quot;Šírka 1\2&quot;&quot;`. [19]



## Cross-Site Request Forgery (CSRF)

Cross - Site Request Forgery je útok spočívajúci v prinútení užívateľa navštíviť stránku, ktorá tajne vykoná útok na webovú aplikáciu, kde je používateľ práve prihlásený. Možno takto napríklad pozmeniť alebo zmazať článok bez vedomia užívateľa. Proti útoku sa možno brániť generovaním a overovaním autorizačného tokenu. Proti tomuto útoku je možné chrániť formuláre v Nette automaticky zavolaním príkazu `$form->addProtection();`. [19]

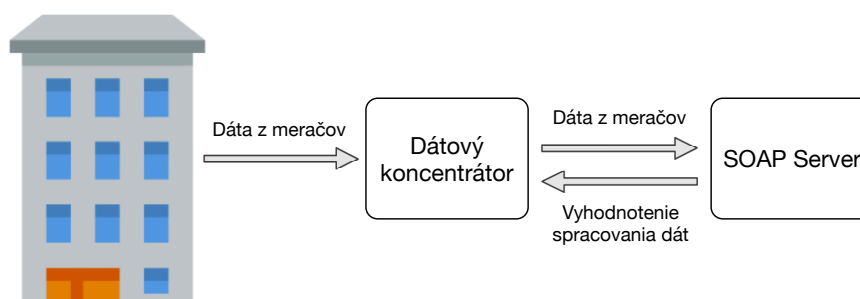
## Ostatné zabezpečenie

Medzi ďalšie časté útoky na aplikácie patria napríklad URL attack, control codes, invalid UTF-8. Cieľom útočníka je podstrčiť vašej webovej aplikácii škodlivý vstup. Následky môžu byť veľmi rôznorodé, od poškodenia XML výstupov cez získanie citlivých informácií z databázy alebo hesiel. Obranou je dôkladné ošetrenie všetkých vstupov, ktoré Nette prevádza automaticky a nie je potrebné nič nastavovať.

Podobnú automatickú ochranu Nette poskytuje proti útokom typu hijacking, session stealing, session fixation, pri ktorých útočník buď scudzí alebo podstrčí užívateľovi svoje session ID a vďaka tomu získa prístup do webovej aplikácie, bez toho, aby poznal heslo používateľa. Obrana spočíva v správnej konfigurácii serveru a PHP. O tú sa Nette postará automaticky v prípade, že je povolená funkcia `ini_set()`. [19]

## 3 SOAP SERVER

V tejto kapitole je popísaný návrh a vývoj SOAP serveru. Cieľom SOAP serveru je prijať dáta odoslané klientom ( dátový koncentrátor), spracovať ich a následne uložiť do databázy. Tieto dáta popisujú údaje o meraniach, ktoré prebehli na meračoch energií zapojených v dátovom koncentrátore. Po spracovaní údajov server odošle klientovi informáciu o úspešnom, prípadne neúspešnom spracovaní dát s popisom chyby.



Obr. 3.1: Komunikácia medzi SOAP serverom a klientom

Základné požiadavky na SOAP server boli zhrnuté do nasledujúcich bodov:

- spracovanie údajov zasielaných klientom a ich ukladanie do databázy
- odoslanie odpovedi klientovi po spracovaní dát
- odosielanie a ukladanie hlásení o chybách vzniknutých pri prenose a spracovaní dát
- možnosť jednoduchšej konfigurácie SOAP serveru

Pri návrhu serveru boli k dispozícii definície prenášaných dát vo formáte XML a popis metód, ktoré volajú klienti vo WSDL schéme. Na základe týchto údajov bola vykonaná analýza a návrh serveru popísaný nasledujúcich sekciách.

### 3.1 Analýza dát a metód

#### 3.1.1 Metóda DataService

Na základe WSDL popisu poskytnutého firmou Modemtec boli identifikované metódy webových služieb, ktoré je nutné implementovať v SOAP servery. Jediná metóda, ktorú je nutné zohľadniť je v tomto prípade len **DataService**. Táto metóda

slúži na prenášanie nameraných údajov medzi klientom a serverom a taktiež na odoslanie spätnej väzby klientovi. Jej definíciu pomocou WSDL môžete vidieť vo výpise kódu č. 3.1.

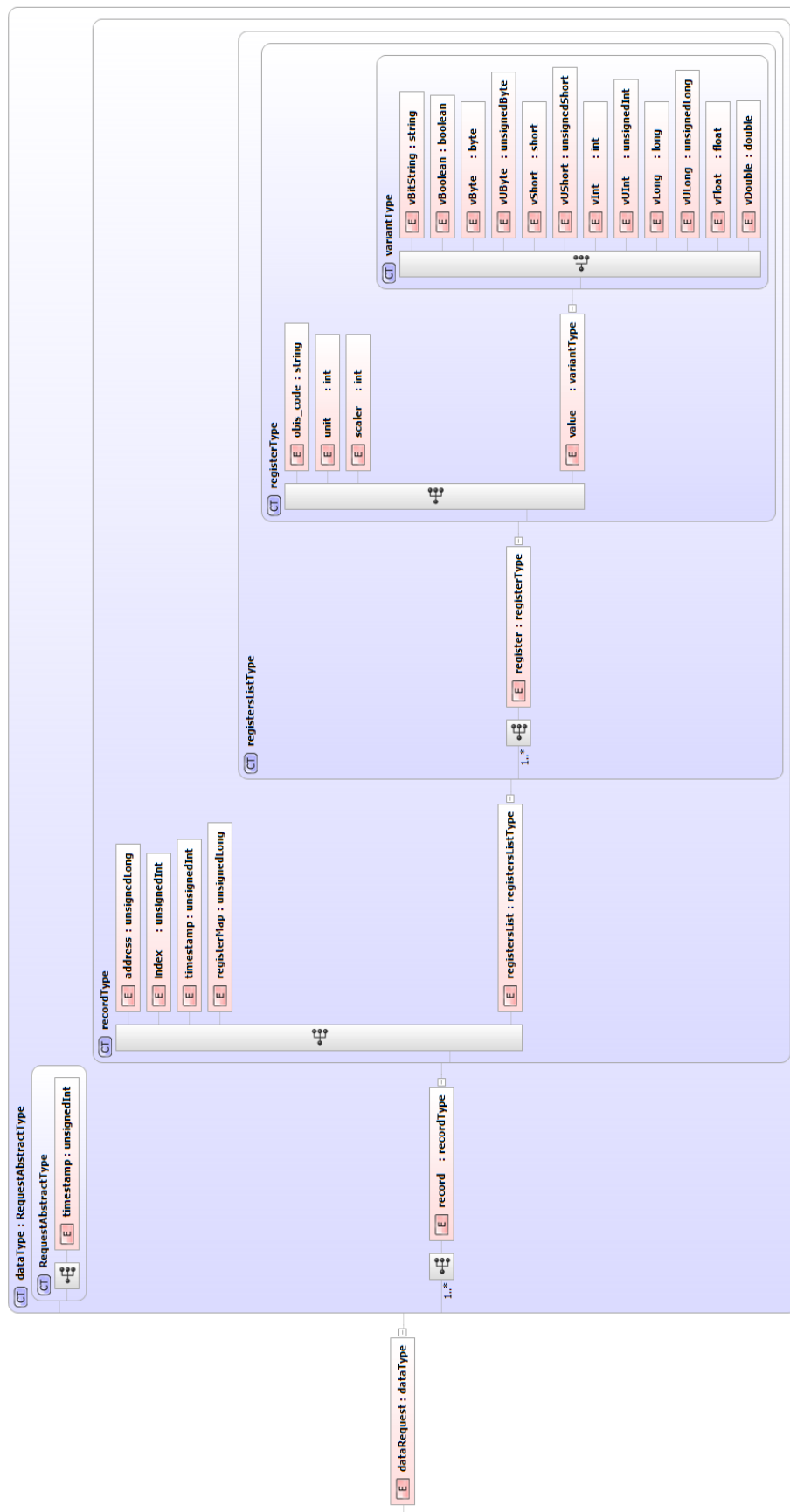
Výpis kódu 3.1: Definícia metódy DataService

```
1 <wsdl:message name="DataService">
2   <wsdl:part name="parameters" element="dataRequest"/>
3 </wsdl:message>
4 <wsdl:message name="DataServiceResponse">
5   <wsdl:part name="parameters" element="dataResponse"/>
6 </wsdl:message>
```

Z definície je zrejmé, že metóda využíva dva dátové typy - `dataRequest` pri jej volaní a `dataResponse` pri odosielaní odpovede.

### 3.1.2 Dátový typ dataRequest

Dátový typ `dataRequest` je definovaný v XML formáte a slúži na popis informácií o vykonanom meraní. Obsahuje údaje, ktoré identifikujú merač na ktorom bolo meranie vykonané, zoznam typov nameraných údajov a aj konkrétne odčítané hodnoty. Vizualizáciu tohoto dátového typu na základe jeho XML popisu môžete vidieť na nasledujúcom obrázku.



Obr. 3.2: Dátový typ dataRequest

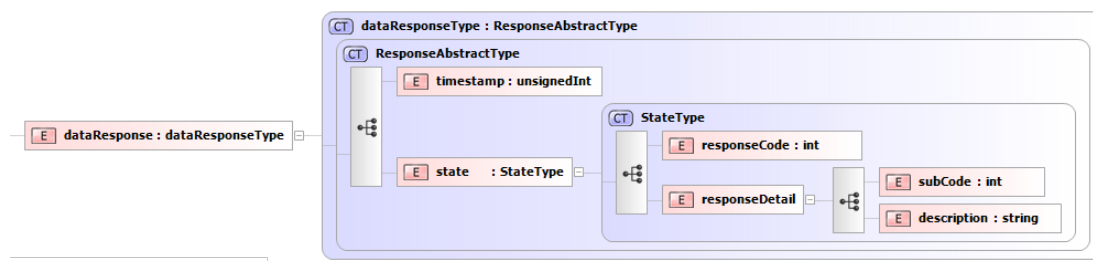
V tabuľke č. 3.1 môžete vidieť popis jednotlivých premenných používaných v dátovom type `dataRequest`. V tabuľke sa nenachádza popis premenných `index` a `scaler`, ktoré na základe požiadavky od firmy Modemtec pri navrhovaní systému neboli brané do úvahy. Pre zjednodušenie je v tabuľke uvedený popis konkrétnych premenných namiesto dátových typov, v ktorých sú zabalené a dátové štruktúry sú označené žltým. Pri premennej `value` je vynechaný popis premenných, ktoré obsahuje. Jedná sa o možné dátové typy nameranej hodnoty (`vBitString`, `vBoolean`, atď) a ich názov by mal byť samovysvetľujúci.

Tab. 3.1: Popis premenných v dátovom type `dataRequest`

Názov	Popis
record	Meranie na prístroji
timestamp	Čas merania
address	Unikátny identifikátor merača
registerMap	Mapa registrov
registerList	Zoznam dátových typov(registrov) v danom meraní
register	Konkrétny dátový typ
obis_code	Identifikátor typu meranej hodnoty
unit	Meraná jednotka
value	Konkrétna nameraná hodnota

### 3.1.3 Dátový typ `dataResponse`

Podobne ako dátový typ `dataRequest` aj dátový typ `dataResponse` je definovaný v XML formáte. Tentokrát sa jedná o jednoduchšiu štruktúru slúžiacu na odoslanie spätnej väzby klientovi po spracovaní údajov. Obsahuje čas spracovania záznamu, stavové kódy a popis spracovania dát. Vizualizáciu tohoto dátového typu na základe jeho XML popisu môžete vidieť na obrázku č. 3.3.



Obr. 3.3: Dátový typ `dataResponse`

V nasledujúcej tabuľke môžete vidieť popis jednotlivých premenných používaných v dátovom type `dataResponse`. Pre tabuľku platia rovnaké pravidlá ako pri dátovom type `dataRequest`.

Tab. 3.2: Popis premenných v dátovom type `dataRequest`

Názov	Popis
timestamp	Čas spracovania dát
state	Stav spracovaných dát
responseCode	Kód odpovede
responseDetail	Bližší popis spracovania
subCode	Kód upresňujúci stav
description	Popis stavu

## 3.2 Databázový model pre namerané dáta

Databázový model popisujúci jednotlivé merania bol vytvorený, aby čo najefektívnejšie splnil svoj účel a bol jednoducho rozširiteľný o ďalšie atribúty. V databáze je pomocou relácií a referenčnej integrity kontrolované, aby nedošlo k zmazaniu dát na ktoré sú pripojené ďalšie záznamy.

Časť dátového modelu bola prebratá od spoločnosti Modemtec, konkrétne sa jedná časť označovanú ako Driver, ktorú je možné vidieť na obrázku č. 3.4 ohraničenú zeleným pozadím. V tejto časti sa nachádza popis prístrojov, dátových typov a meračov. Pri návrhu SOAP serveru bolo nutné vytvoriť prepojenie na Driver, aby bolo možné pri spracovávaní nameraných dát získať podrobnejšie údaje o prístrojoch.

### 3.2.1 Identifikácia entít

V nasledujúcej tabuľke môžete vidieť popis jednotlivých entít. Za entitu je považovaná množina údajov spájajúca dáta patriace k sebe na základe logických súvislostí. Pre prehľadnosť uvádzam entity oddelene, pre časť modelu popisujúcu namerané dáta a časť modelu pre driver.

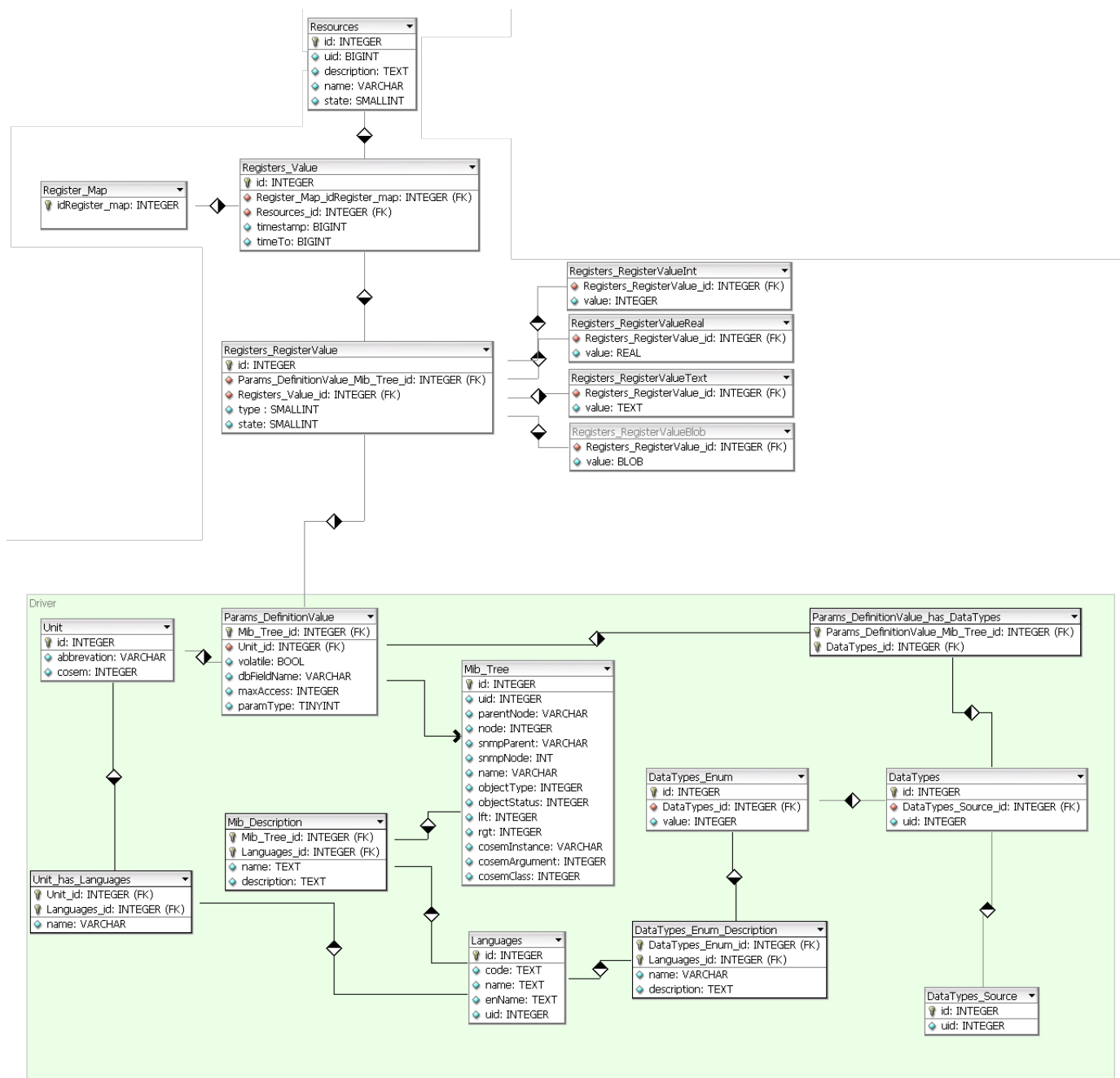
Tab. 3.3: Popis entít v databázovom modeli pre namerané dáta

Názov	Popis entity
Resources	Merače
Registers_Value	Merania na merači
Registers_RegisterValue	Jedna hodnota obsiahnutá v meraní
Register_Map	Mapa registrov
Registers_RegisterValueReal	Nameraná hodnota typu REAL
Registers_RegisterValueInt	Nameraná hodnota typu INT
Registers_RegisterValueText	Nameraná hodnota typu TEXT
Registers_RegisterValueBlob	Nameraná hodnota typu BLOB

Tab. 3.4: Popis entít v databázovom modeli pre driver

Názov	Popis entity
Unit	Tabuľka pre popis jednotiek registrov
Params_DefinitionValue	Rozšírená definícia parametrov stromu
DataTypes	Zoznam dátových typov pre všetky zdroje
DataTypes_Source	Zoznam zdrojov v rámci, ktorých sú používané jednotlivé dátové typy(SNMP, SOAP, SQLite, Visualization)
DataTypes_Enum_Description	Názvy jednotlivých hodnôt enumerovaných dátových typov
DataTypes_Enum	Definovanie jednotlivých hodnôt pre enumerované dátové typy alebo bitové mapy
Languages	Všetky podporované jazyky
Mib_Tree	Definovanie MIB stromu pre jednotlivé objekty
Mib_Description	Popis MIB stromu

### 3.2.2 ER diagram



Obr. 3.4: ER diagram pre namerané dáta



## 3.3 Vývoj SOAP server

### 3.3.1 Použité technológie

SOAP server bol vytvorený pomocou programovacieho jazyka PHP a HTTP servera bežiacom na Apache, ktorý sa staral o komunikáciu. Pri tvorbe bol využitý objektový prístup a štandardné triedy implementované v jazyku PHP, konkrétne:

- SoapServer Class - PHP trieda poskytujúca funkčnosť serveru pre protokol SOAP 1.1 a SOAP 1.2 umožňujúcu využívať WSDL popis webových služieb.
- PDO Class - PHP trieda reprezentujúca komunikáciu medzi serverom a databázou.

### 3.3.2 Adresárová štruktúra

V nasledujúcej tabuľke sa nachádza adresárová štruktúra vytvorená pre SOAP server.

Tab. 3.5: Súborová štruktúra SOAP server

Názov	Popis
index.php	Načítanie konfigurácie, nastavenie servera a spracovanie požiadaviek
DataClass.php	Implementácia metód poskytovaných serverom, spracovanie dát a ich ukladanie do databázy
config.ini	Konfigurácia databázy a ukladania chýb
schemas/	XML popisujúce prenášané dáta
wSDL/	WSDL popisujúce webovú službu

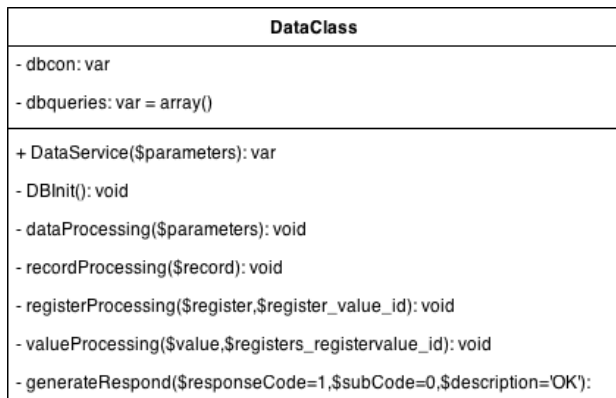
### 3.3.3 Realizácia

Pri programovaní SOAP serveru bol kladený dôraz na vytvorenie jednoduchého základu, ktorý umožní rozširovanie funkčnosti a rýchlu úpravu serveru v prípade zmeny formátu dát odosielaných klientmi.

Pri implementácii je využitý objektový prístup a odchytyvanie výnimiek. Samostatný SOAP server, ktorý má na starosti prenos dát je vytvorený v `index.php` pomocou príkazu `$server=new SoapServer('wSDL/DataService.wSDL')` a je mu priradená trieda `DataClass` `$server->SetClass('DataClass')`. Trieda `SoapServer($server)` od tohto momentu rieši predávanie dát medzi serverom a klientom. Tieto dáta sú spracovávané triedou `DataClass`, ktorej popis sa nachádza nižšie.

## Trieda DataClass

Trieda `DataClass` slúži na spracovanie prijatých dát typu `dataRequest` a na vytvorenie spätnej väzby pre klientov typu `dataResponse`. V tejto triede je vytvorené aj pripojenie k databáze a následné uloženie spracovaných dát. Atribúty a metódy implementované v `DataClass` je možné vidieť na obrázku č. 3.6.



Obr. 3.5: UML model triedy `DataClass`

Jediná verejne prístupná metóda je `DataService($parameters)`, ktorá reprezentuje metódy webových služieb popísanú vo WSDL. Táto metóda je automaticky volaná SOAP serverom, keď príde požiadavka od klienta. Jej vstupným argumentom je premenná `$parameters`, tá obsahuje informácie o meraní popísané dátovými objektami typu `stdClass`. Jedná sa o generickú triedu definovanú v PHP na ktorú sú prevedené XML dáta typu `dataRequest` pri spracovávaní triedou `SoapServer`. Ukážku takýchto dát môžete vidieť na výpise kódu č. 3.2.

Výpis kódu 3.2: Dáta odoslané klientom spracované triedou `SoapServer Class`

```
1 stdClass::__set_state(array(
2     'timestamp' => 0,
3     'record' =>
4     stdClass::__set_state(array(
5         'address' => 987654321,
6         'index' => 0,
7         'timestamp' => 1426531038,
8         'registerMap' => 120,
9         'registersList' =>
10        stdClass::__set_state(array(
11            'register' =>
12            stdClass::__set_state(array(
```

```

13         'obis_code' => '7.0.3.0.0.255',
14         'unit' => 255,
15         'scaler' => 0,
16         'value' =>
17         stdClass::__set_state(array(
18             'vUShort' => 579,
19         )),
20     )),
21 )),
22 ));

```

Samostatná metóda slúži ako iniciátor spracovávania dát. Najskôr je vytvorené pripojenie k databáze a potom sú dáta odoslané privátnej metóde `dataProcessing(\ $parameters)`, ktorá dáta postupne spracováva pomocou ostatných metód, konkrétne `recordProcessing`, `registerProcessing` a `valueProcessing`. V tých metódach sú dáta postupne extrahované z premennej `$parameters` a nahrávané do databázy. Pri ukladaní dát do databázy sú využívané pred-pripravené SQL dotazy uložené v atribúte `dbqueries`. Všetky SQL dotazy sú uložené na jednom mieste a je ich jednoduché upravovať a zároveň sa zmenší ich celkový počet. Príklad predpripraveného dotazu a jeho volanie môžete vidieť na výpise kódu č. 3.3.

Výpis kódu 3.3: Predpripravené SQL dotazy a ich volanie

```

1  //Pripravenie dotazu
2  $query ="INSERT INTO registers_registervalue (
        registers_registervalue_id,value) VALUES (:
        registers_registervalue_id,:data);";
3  $this->dbqueries['insert_registervalue'] = $this->
        dbcon->prepare($query, array(PDO::ATTR_CURSOR => PDO::
        CURSOR_FWDONLY));
4
5  //Volanie dotazu
6  $data=5.4;
7  $this->dbqueries['insert_registervalueint']->execute(array
        (':registers_registervalue_id' =>
        $registers_registervalue_id, ':data' => $data));

```

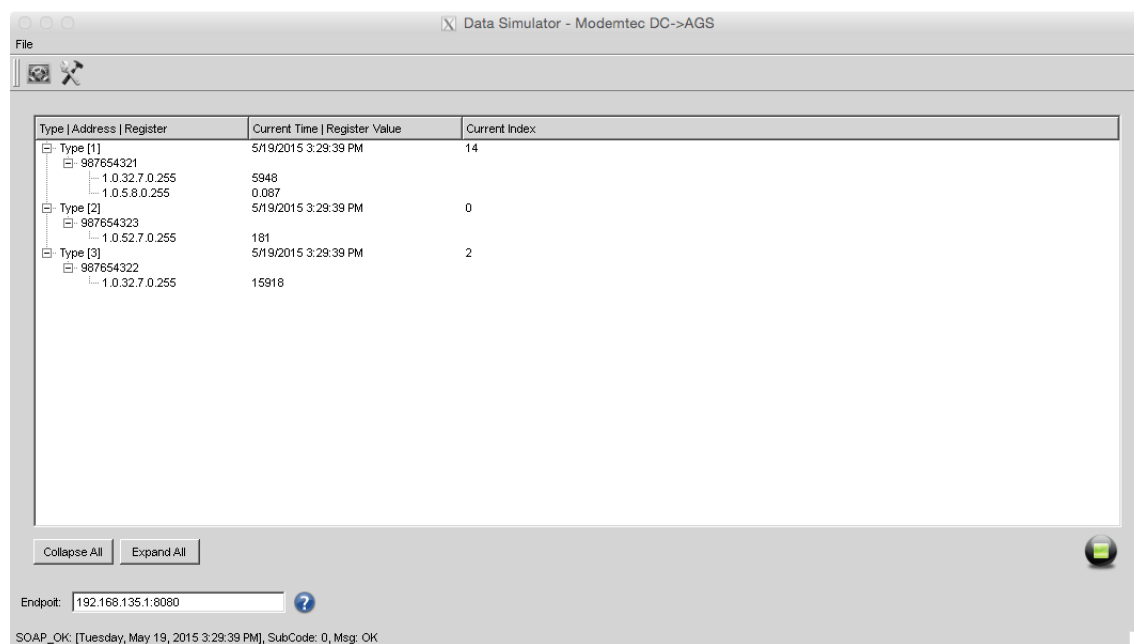
Po ukončení spracovávania je zavolaná metóda `generateRespond`, ktorá odošle klientovi kód a popis potvrdzujúci úspešné spracovanie informácií. V prípade vzniku chyby pri spracovávaní alebo inicializácii je vytvorená výnimka, ktorá je zachytená v triede `DataClass`. Dôvodom na vznik chyby môže byť nemožnosť sa pripojiť k databáze, poprípade nesprávny formát dát. Ak podobná situácia nastane je spracovanie prerušené, chyba je zaznamenaná v logoch a klientovi je odoslaný chybový

kód s popisom.

### 3.3.4 Testovanie

Testovanie vytvoreného SOAP serveru prebiehalo pomocou aplikácie Data Simulator od firmy Modemtec. Táto aplikácia vytvára simuláciu meraní na meračoch prebiehajúcu v reálnom čase a namerané dáta odosiela na server s využitím protokolu SOAP.

Cieľom testovania bolo zistiť, či server posielať dáta zachytí a spracuje. Pri testovaní boli použité rôzne kombinácie dátových typov a rôzne intervaly odosielania chybových hodnôt. Pri simuláciách boli úmyselne nastavované nezmyselné a neexistujúce údaje, aby bola preverená schopnosť serveru reagovať na poruchy a logovanie týchto chýb. Dáta, ktoré boli spracované serverom boli ručne kontrolované, aby bola preverená ich správnosť. Pri testovaní servera bola objavená chyba so spracovaním dátového typu REAL, ktorá bola následne opravená.



Obr. 3.6: Simulátor odosielania nameraných dát pomocou protokolu SOAP

## 4 NÁVRH INFORMAČNÉHO SYSTÉMU

V tejto kapitole môžete nájsť, kroky ktoré boli vykonané pred vývojom informačného systému. Konkrétne sa jedná o stanovovanie všeobecných požiadaviek, ktoré by mal systém spĺňať, definovanie funkcionality systému a vytvorenie diagramov prípadov použitia pre jednotlivých aktérov v systéme. Na základe týchto informácií boli následne navrhnuté entity a vytvorený ER diagram popisujúci dátový model.

### 4.1 Analýza požiadaviek

#### 4.1.1 Všeobecné požiadavky

Cielom je vytvoriť informačný systém zobrazujúci namerané dáta z meračov energií uložené v databáze. Webový systém má byť kompatibilný s najpopulárnejšími webovými prehliadačmi a má byť použiteľný aj na mobilných telefónoch. Súčasťou systému musí byť administrátorské rozhranie umožňujúce správu systému a užívateľov. Je nutné oddeliť obsahovú stránku informačného systému od vizuálnej stránky, predpokladá sa využitie architektúry MVC.

#### 4.1.2 Rozšírené funkčné požiadavky

V nasledujúcich bodoch je zhrnutá základná funkcionality informačného systému:

- **Užívateľské prostredie** prehľadne zobrazujúce zoznam budov, bytov a meračov automaticky sa prispôbujúcich podľa oprávnení užívateľa.
- **Spracovanie nameraných údajov** s podporou filtrov
  - Tabuľka, zobrazujúca namerané dáta
  - Graf
  - Filter údajov podľa dátumu
- **Autentifikácia užívateľov**
  - Prihlásenie
  - Obnovenie hesla
- **Administrácia meračov**
  - Zoznam meračov
  - Odoberanie meračov
  - Editácia meračov
  - Vyhľadávanie merača
- **Administrácia užívateľov**
  - Zoznam užívateľov
  - Pridávanie užívateľa

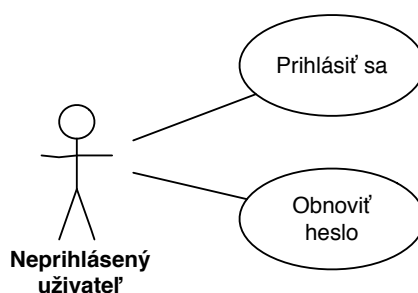
- Odoberanie užívateľa
- Editácia užívateľa
- Vyhľadávanie užívateľa
- **Administrácia dodávateľov**
  - Zoznam dodávateľov
  - Pridávanie dodávateľa
  - Odoberanie dodávateľa
  - Editácia dodávateľa
  - Vyhľadávanie dodávateľa
- **Editácia osobných údajov a nastavení užívateľov**

### 4.1.3 Diagramy prípadov použitia

V tejto časti je pomocou krátkeho popisu a diagramov prípadov použitia zobrazená funkčnosť systému pre jednotlivých aktérov v systéme. Pojmom aktér je myslený užívateľ komunikujúci so systémom, v tomto prípade konkrétne : neprihlásený užívateľ, zákazník, dodávateľ a administrátor.

#### Neprihlásený užívateľ

Neprihlásený užívateľ nemá právo využívať systém a má prístup len k prihláseniu do systému a obnoveniu hesla.



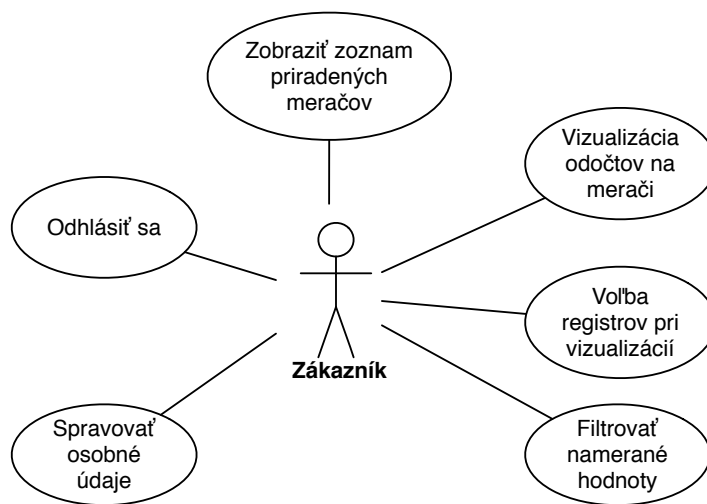
Obr. 4.1: Diagram prípadov použitia - Neprihlásený užívateľ

#### Zákazník

Zákazník je užívateľ vlastniaci jeden alebo viac bytov. Využíva informačný systém na zobrazenie spotreby energií v jednotlivých bytoch.

Má nasledujúce oprávnenia:

- Zobrazenie údajov z jednotlivých meračov
- Editácia osobných údajov a nastavení



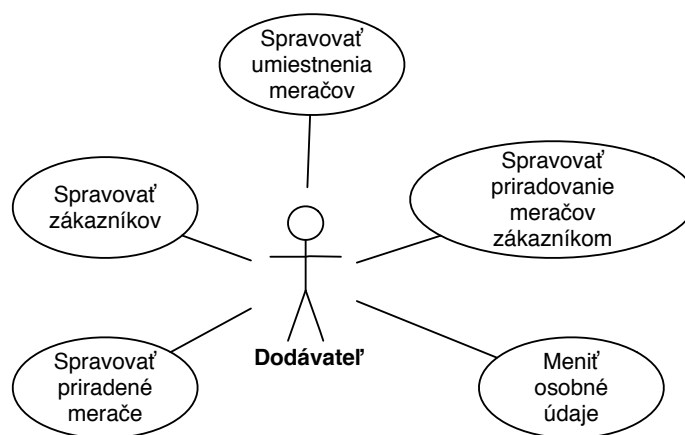
Obr. 4.2: Diagram prípadov použitia - Zákazník

## Dodávateľ

Dodávateľ je užívateľ poskytujúci zákazníkovi dodávku energií a správu jednotlivých meračov. Využíva informačný systém na administráciu svojich zákazníkov, meračov a zobrazenie nameraných hodnôt.

Má nasledujúce oprávnenia:

- Zobrazenie údajov z jednotlivých meračov
- Administrácia meračov
- Administrácia užívateľov
- Editácia osobných údajov a nastavení



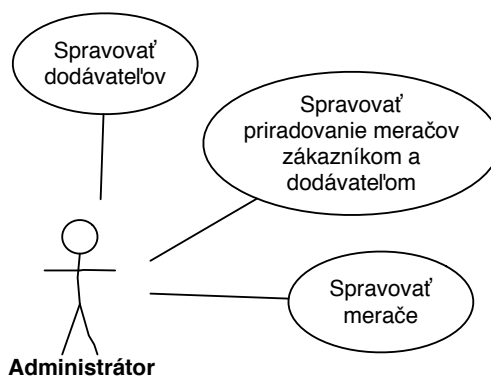
Obr. 4.3: Diagram prípadov použitia - Dodávateľ

## Administrátor

Administrátor je užívateľ spravujúci kontá zákazníkov a dodávateľov. Využíva informačný systém na administráciu celého systému.

### Má nasledujúce oprávnenia:

- Zobrazenie údajov z jednotlivých meračov
- Administrácia meračov
- Administrácia užívateľov
- Administrácia dodávateľov
- Editácia osobných údajov a nastavení



Obr. 4.4: Diagram prípadov použitia - Administrátor



## 4.2 Návrh databázového modelu pre informačný systém

V tejto sekcii je popísaný postup pri návrhu databázového modelu pre informačný systém. Jedná sa o rozšírenie databázového modelu z kapitoly č. 3 o prvky obsahujúce údaje o užívateľoch a polohe meračov.

### 4.2.1 Identifikácia entít

Entita je množina dát spájajúca dáta patriace k sebe na základe logických súvislostí. V nasledujúcej tabuľke sú popísané entity použité v ER diagrame.

Tab. 4.1: Popis entít v databázovom modeli pre informačný systém

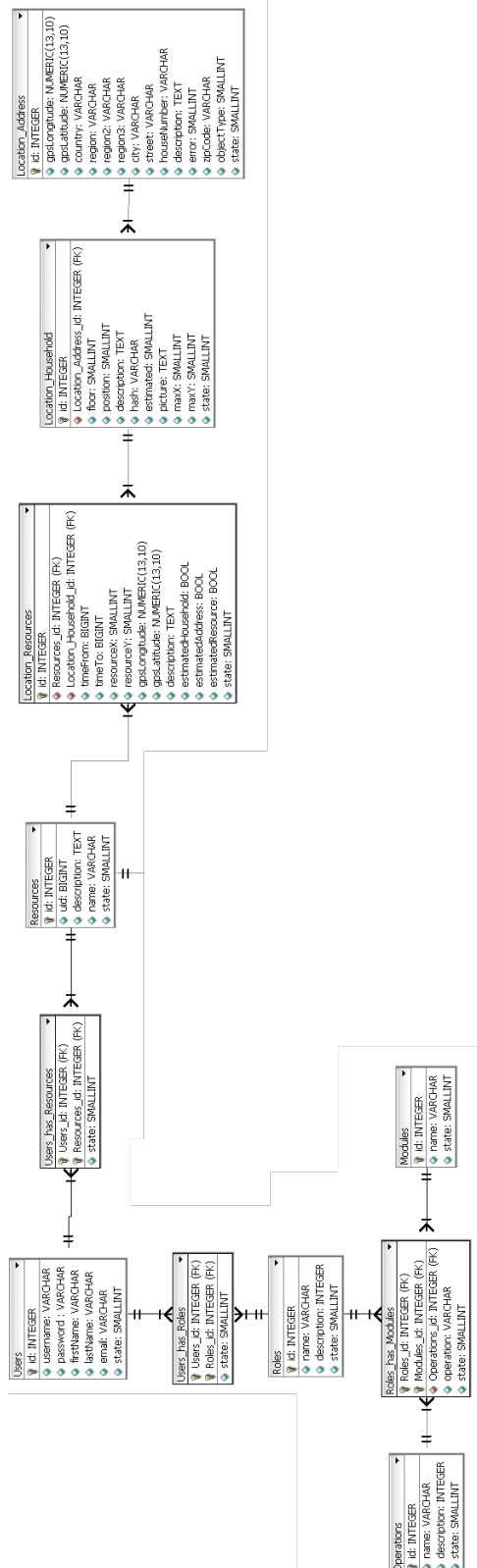
Názov	Popis entity
Users	Užívatelia
Roles	Užívateľské role
Operations	Operácie, ktoré je možné vykonávať nad modulmi - napr. čítanie, zápis, editácia
Modules	Moduly informačného systému
Resources	Merače
Location_Resources	Poloha merača v domácnosti
Location_Household	Domácnosť
Location_Address	Adresa

### 4.2.2 ER diagram

Pri tvorbe ER diagramu pre informačný systém boli určené vzťahy medzi jednotlivými entitami a na ich základe boli vytvorené väzobné tabuľky pre väzbu typu M ku N.

Tab. 4.2: Zoznam väzobných tabuliek

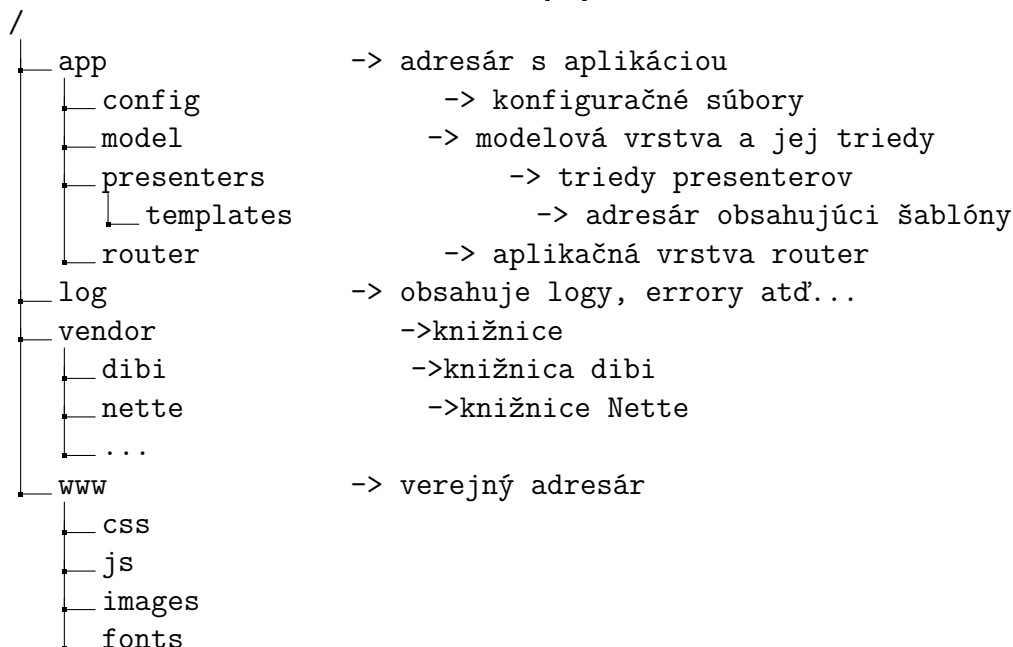
Názov	Popis
Users_Has_Resources	Merače priradené užívateľom
Users_Has_Roles	Role priradené užívateľom
Roles_Has_Modules	Moduly priradené roliam



Obr. 4.5: ER diagram informačného systému

## 4.3 Adresárová štruktúra systému

Štruktúra systému bola navrhnutá na základe doporučenej adresárovej štruktúry odporúčenej tvorcami Nette frameworku. [18]. Jej popis môžete vidieť nižšie.



## 4.4 Architektúra systému

Celý systém je navrhnutý s ohľadom na zabezpečenie informačného systému a s ohľadom na jednoduchú rozšíriteľnosť. Systém je postavený na PHP Frameworku Nette a je pri jeho návrhu využívaný prístup MVC(Model-View-Controller), ktorý je bližšie popísaný v teoretickom úvode v sekcii č. 3. V tejto sekcii sa nachádza popis jednotlivých aplikačných vrstiev a tried, ktoré boli vytvorené.

### 4.4.1 Model

Aplikačná vrstva Model reprezentuje komunikáciu medzi informačným systémom a databázou. Je rozdelený na tri samostatné triedy podľa typu operácií, ktoré vykonáva.

#### UserManager

Táto trieda slúži na správu užívateľov. Obsahuje metódy používané na overenie prihlásenia, správu užívateľov a ich odstránenie. Trieda využíva hashovacie mechanizmy implementované v Nette na bezpečné ukladanie a overenie hesla. Je zároveň použitá na vytvorenie a predanie identity užívateľa pri jeho prihlásení.

## **RoleAuthorizator**

Táto trieda je implementácia ACL (Access Control List) a slúži na kontrolu oprávnení jednotlivých užívateľov. Je založená na triede **IAuthorizator**, ktorá je poskytovaná frameworkom Nette. Jej funkčnosť je rozšírená o prácu s dátami uloženými v data-báze a je plne kompatibilná s metódami implementovanými v Nette.

## **ResourceManager**

Hlavnou úlohou **ResourceManager** je pracovať s dátami o meračoch. Obsahuje metódy na správu meračov, lokácii, domácností a adries. Je využívaná takmer v celom systéme pri upravovaní a pridávaní dát.

### **4.4.2 Presentery**

Triedy typu presenter predstavuje aplikačnú vrstvu controller a slúži na spracovávanie dát získavaných z modelu. Presentery predstavujú jednotlivé moduly informačného systému s výnimkou **BasePresenter** a **PublicPresenter**, ktoré slúžia ako abstraktné triedy kontrolujúce oprávnenia.

#### **BasePresenter**

**BasePresenter** je abstraktná trieda, ktorá je základom (predkom) pre väčšinu ostatných presenterov. Je neprístupná neprihláseným užívateľom a obsahuje metódy, ktoré overujú práva užívateľov na zobrazenie obsahu na ktorom sa práve nachádzajú. V prípade, že užívateľ nie je prihlásený, alebo nemá dostatočné práva na vykonanie operácie je automaticky presmerovaný.

Sú v nej definované aj pomocné metódy na zjednodušenie práce v ostatných prezenteroch, ktoré slúžia na kontrolovanie zadaných vstupov a overenie správnosti URL. V **BasePresenter** sú taktiež spracovávané dáta, ktoré sa týkajú stabilnej časti užívateľského prostredia ako je menu a nastavenie východzieho pohľadu.

#### **PublicPresenter**

**PublicPresenter** je abstraktná trieda, ktorá je základom pre presentery, ktoré sú prístupné verejnosti. V tomto prípade sa jedná len o **SignInPresenter** a **ErrorPresenter**, ktoré slúžia na prihlásenie užívateľov a zobrazenie chybových hlášok.

#### **Homepage**

V tejto triede sú spracovávané údaje zobrazené užívateľovi pri prihlásení do systému. Jedná sa o celkový prehľad meračov, ktoré sú užívateľovi prístupné.

## **ResourcePresenter**

V **ResourcePresenter** sú spracovávané dáta o konkrétnom merači, ktoré sú použité na grafické zobrazenie nameraných údajov.

## **AdminUsersPresenter**

Úlohou **AdminUserPresenter** je administrácia užívateľov. Obsahuje formuláre a metódy na administráciu zákazníkov a dodávateľov. Jeho súčasťou sú moduly na priradenie oprávnení na prístup k údajom o jednotlivých meračoch užívateľom.

## **AdminResourcePresenter**

Tento presenter zabezpečuje spracovanie údajov o meračoch a ich kompletnú administráciu. Obsahuje formuláre potrebné na editovanie a mazanie meračov. Taktiež na administráciu lokácií meračov.

## **SettingsPresenter**

Jedná sa o triedu, ktorá v systéme umožňuje užívateľom meniť svoje osobné nastavenia.

## **ErrorPresenter**

V prípade poruchy spracuje chybové stavy a zobrazí užívateľovi prijateľnú hlášku.

## **SignPresenter**

Zabezpečuje volanie metód z moduly pri prihlasovaní užívateľa. V prípade úspešného prihlásenia založí užívateľovi identitu a povolí jeho prístup do systému.

### **4.4.3 Šablóny**

Šablóny slúžia v systéme na vykreslenie užívateľského rozhrania. Každá šablóna je priradená k presenteru, ktorý spracováva dáta z modelu, ktoré sú následne v šablóne zobrazené. Vývoj šablón bol zameraný na maximálnu prehľadnosť a jednoduchosť zdrojového kódu, aby bola zvýšená jeho znovu-použiteľnosť.

Obrazovky využívajú hlavnú šablónu na zobrazenie obsahu, ktorý sa nemení. V nette sa táto šablóna nazýva layout a zabezpečuje rozloženie prvkov v informačnom systéme, zobrazenie užívateľského menu a podobne. Časť, ktorá sa pre každú stránku mení je vytvorená samostatnou šablónou, ktorá je priradená k presenterom. V nasledujúcej tabuľke sú popísané šablóny, ktoré sú v systéme vytvorené a prezentujú jednotlivé užívateľské obrazovky.

Tab. 4.3: Zoznam šablón

Homepage:	
default	Úvodná stránka zobrazujúca zoznam meračov
ResourcePresenter:	
default	Vizualizácia nameraných dát a ich výpis
AdminUserPresenter:	
customerAdd	Pridávanie nových zákazníkov
customerEdit	Editácia zákazníkov
customerHasResources	Priradovanie meračov k zákazníkom
customersGrid	Zoznam zákazníkov
providerAdd	Pridávanie nových dodávateľov
providerEdit	Editácia dodávateľov
providerHasResources	Priradovanie meračov k dodávateľom
providersGrid	Zoznam dodávateľov
AdminResourcePresenter:	
addressAdd	Pridávanie novej adresy
addressEdit	Editácia existujúcej adresy
householdAdd	Pridávanie novej domácnosti
householdEdit	Editácia domácnosti
resourceEdit	Editácia merača
resourceGrid	Zoznam meračov
resourceLocationAdd	Pridávanie novej polohy merača
resourceLocationEdit	Editácia polohy merača
SettingPresenter:	
default	Zobrazenie a upravovanie nastavení užívateľa
ErrorPresenter:	
default	Zobrazenie chybovej hlášky
SignPresenter:	
in	Formulár na prihlásenie užívateľa

## 4.5 Realizácia

### 4.5.1 Zabezpečenie proti nepovolenému vstupu

Informačný systém je chránený proti nepovolenému vstupu a pre prístup k väčšine obrazoviek je nutné byť prihlásený. Ak užívateľ otvorí obrazovku, ktorá nie je verejne prístupná a nie je prihlásený je automaticky presmerovaný na prihlasovaciu stránku,

kde po vyplnení údajov je zavolaná metóda `authenticate(array $credentials)`. Táto metóda slúži na autentifikáciu prihlasovacích údajov a kontroluje zhodu hesla uloženého v databáze so zadaným heslom v prihlasovacom formulári. Heslá sú z bezpečnostných dôvodov uložené v databáze v zašifrovanej podobe. O zabezpečenie sa stará framework Nette pomocou `Passwords::hash($password)`. Táto metóda využíva technológiu salted hash, ktorá k heslu pridá náhodnú postupnosť znakov (nazývanú salt), heslo zahashuje a uloží heslo do databáze.

Po úspešnom prihlásení je užívateľovi vygenerovaná identita, ktorá je uložená v globálnej premennej `$_SESSION`. V tejto identite sú uložené základné údaje ako je prihlasovacie meno, email a zoznam rolí.

Ak je užívateľ úspešne prihlásený je potrebné overiť, či má prístup k operáciám o ktorú požiadal. O kontrolu oprávnení sa stará metóda `isAllowed($resource,$privilege)`, ktorá prejde všetky role, ktoré má užívateľ pridelené a overí, či má k operácii prístup. Ak je zistený, že užívateľ nemá oprávnenie je automaticky presmerovaný na domovskú stránku, kde mu je zobrazená informačná hláška.

## 4.5.2 Formuláre

Veľká časť informačného systému je tvorená obrazovkami, ktoré umožňujú dáta editovať alebo vypisovať. Pri tvorbe formulárov je využívaná trieda implementovaná v Nette konkrétne `Nette\Application\UI\Form`. Trieda `Form` zabezpečuje vytvorenie formulára a zároveň zabezpečenie vstupných dát na strane klienta aj serveru. Ak je formulár úspešne vyplnený a odoslaný, je zavolaná definovateľná metóda `presenteru`, ktorá výkonná potrebné operácie nad prijatými údajmi. Príklad použitia tejto triedy môžete vidieť vo výpise kódu č. 4.1

Výpis kódu 4.1: Ukážka použitia triedy `Nette\Application\UI\Form`

```
1 $form = new Nette\Application\UI\Form;
2 $form->addText('username', 'Username:')
3     ->setDisabled(TRUE);
4 $form->addText('firstname', 'Meno:')
5     ->setDisabled(TRUE);
6 $form->addText('lastname', 'Priezvisko:')
7     ->setDisabled(TRUE);
8 $form->addText('email', 'Email:')
9     ->setRequired()
10    ->addRule(Nette\Application\UI\Form::EMAIL, 'Prosím
    zadajte platnú emailovú adresu.');
```

```
11 $form->addSubmit('send', 'Uložiť');
```

```
12 $form->onSuccess[] = array($this, 'settingsFormSucceeded')
    ;
```

Posledným krokom je vytvoriť v šablóne grafický popis formuláru, ktorý rozhoduje o tom ako bude vyzeráť vykreslenie požadovaného formuláru.

### 4.5.3 Tabuľkový výpis dát

Výpis väčšieho množstva dát bol riešený pomocou komponenty Grido, ktorá je veľmi jednoduchá na implementáciu s databázovým systémom Dibi. Grido už v sebe obsahuje základné filtrovania, exportovanie údajov, čo urýchľuje prácu. Zjednodušenú ukážku komponenty Grido môžete vidieť vo výpise kódu č. 4.3

Výpis kódu 4.2: Ukážka použitia komponenty Grido

```
1 protected function createComponentCustomersGrid($name)
2 {
3     //Vytvorenie komponenty Grido
4     $grid = new \Grido\Grid($this, $name);
5     //Prepojenie k modelu
6     $fluent = $this->context->userManager->getUsersInfo(3);
7     //Nastavenie primárneho kľúča a možnosti exportovania
8     $grid->SetModel($fluent)->setPrimaryKey('id')->setExport
        ('');
9     //Pridanie nového stĺpca do tabuľky
10    $grid->addColumnText('firstname', 'Meno')
11        ->setSortable()
12        ->setFilterText();
13    return $grid;
14 }
```

### 4.5.4 Grafické zobrazenie nameraných hodnôt

Na grafické zobrazenie dát sa využíva javascriptová knižnica morris.js, ktorý bol prepojený s komponentom Grido. Dáta, ktoré sú aktuálne zobrazené v tabuľkovom výpise sú exportované a prevedené do formátu JSON pomocou funkcie PHP `json_encode()`. Tým sa zabezpečuje správne zobrazenie dát aj po použití filtrov. Formát JSON slúži na zadávanie bodov, ktoré budú vykreslené v grafe.

Výpis kódu 4.3: Ukážka použitia knižnice morris.js

```
1 <script>
```



```

2 Morris.Line({
3   element: 'graf-meranie',
4   data: {
5     { timestamp: 1432125129, value: 20 },
6     { timestamp: 1432125155, value: 30 },
7     { timestamp: 1432125233, value: 40 },
8     { timestamp: 1432126333, value: 40 },
9     { timestamp: 1432127222, value: 55 }
10  },
11  xkey: 'timestamp',
12  ykeys: ['value'],
13  labels: ['Hodnota vo kW'],
14  lineColors: ['#0b62a4'],
15  smooth: true,
16  resize: true
17 });
18 </script>

```

#### 4.5.5 Ukážky užívateľského rozhrania

Na nasledujúcich obrázkoch sú zobrazené vybrané obrazovky informačného systému. Obrázok č. 4.6 zobrazuje stránku pre prihlásenia do uzavretej časti systému.

The image shows a web form for logging in. At the top is a title 'Prihlásenie'. Below it are two input fields: 'Username' and 'Heslo'. A green button labeled 'Login' is positioned below the password field. Under the button is a link: 'Zabudli ste svoje heslo ? Kliknite sem.'. At the bottom of the form is a green box with the text 'Boli ste úspešne odhlásený.'.

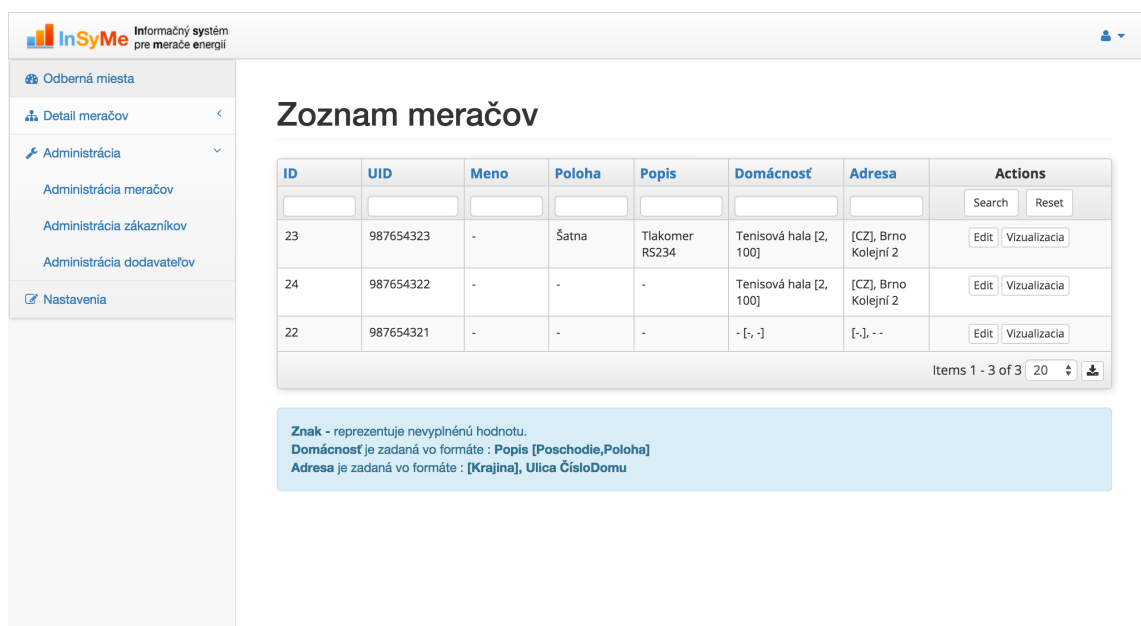
Obr. 4.6: Prihlásenie do systému



Obr. 4.7: Úvodná obrazovka zobrazená prihlásenému užívateľovi

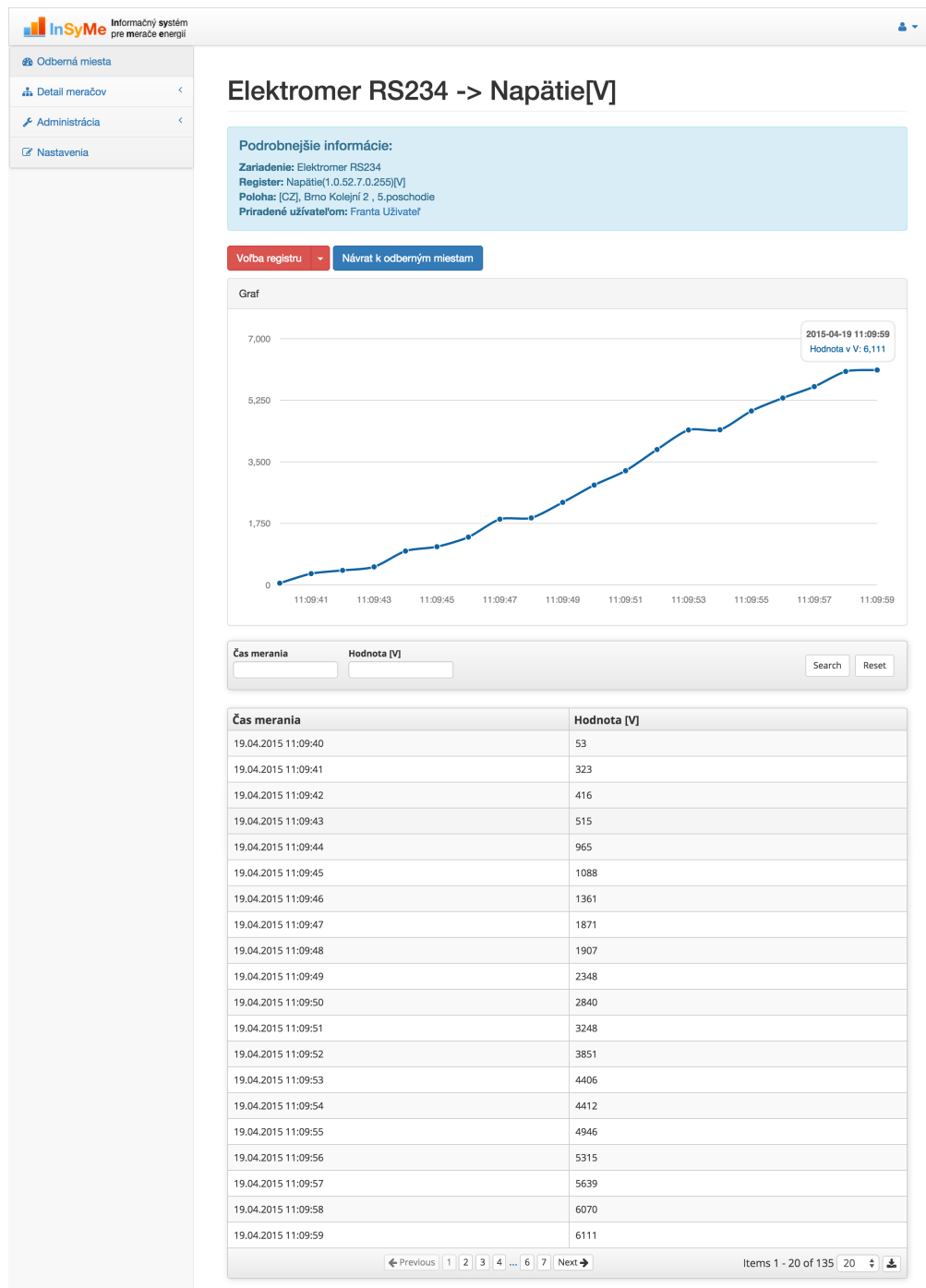
Obrázok vyššie predstavuje úvodnú obrazovku zobrazenú po prihlásení do systému. Na tejto obrazovke sa nachádza zoznam meračov ku ktorým má užívateľ systému prístup .

Nasledujúci obrázok zobrazuje administráciu meračov, ktorá je dostupná správcom systému. Dáta v zozname je možné filtrovať, poprípade zoradiť. Podobné zoznamy sú vytvorené aj na administráciu zákazníkov a dodávateľov.





Obr. 4.8: Administrácia meračov - Zoznam meračov


Na obrázku č. 4.9 sa nachádza grafická a tabuľková reprezentácia nameraných údajov na merači. Je možné si vybrať, ktoré typy údajov posielaných meračom budú zobrazené a dáta sa dajú následne filtrovať.





Obr. 4.9: Detailné zobrazenie konkrétneho merača

 Informačný systém  
pre merače energií

 Odborná miestna

 Detail meračov

 Administrácia

 Nastavenia

## Oprávnenia dodavateľa firmaA[ID:21]

Návrat k zoznamu zákazníkov

Priradené merače

	ID	UID	Meno	Poloha	Popis	Domácnosť	Adresa	Actions
<input type="checkbox"/>								Search Reset
<input type="checkbox"/>	24	987654322	-	-	-	Tenisová hala [2, 100]	[CZ], Brno Koleční 2	Edit Vizualizácia
<input type="checkbox"/>	23	987654323	-	Šatna	Tlakomer RS234	Tenisová hala [2, 100]	[CZ], Brno Koleční 2	Edit Vizualizácia

Selected... Items 1 - 2 of 2 20

Nepriradené merače

	ID	UID	Meno	Poloha	Popis	Domácnosť	Adresa	Actions
<input type="checkbox"/>								Search Reset
<input type="checkbox"/>	22	987654321	-	-	-	- [-, -]	[-, -]	Edit Vizualizácia

Selected... Items 1 - 1 of 1 20

Povinné položky sú označené červeným okrajom.

Návrat k zoznamu zákazníkov

Obr. 4.10: Editácia oprávnení prístupu k meračom

Posledný obrázok zobrazuje časť systému, ktorá umožňuje administrátorom priradovať a odoberať oprávnenia na prístup k meračom.

## 5 ZÁVER

Cielom tejto bakalárskej práce bolo zoznámiť sa s princípmi inteligentných sietí, webových služieb a technológiami potrebnými pre tvorbu informačných systémov a na základe získaných poznatkov navrhnúť a implementovať SOAP server spracúvajúci dáta z meračov a informačný systém pre merače energie.

V prvej kapitole tejto práce boli popísané princípy inteligentných sietí a komponenty nachádzajúce sa v sieti.

V druhej kapitole boli popísané technológie potrebné na vývoj SOAP servera a informačných systémov.

Ďalej nasledovala špecifikácia požiadavok na SOAP server a analýza nameraných údajov zasielaných dátovým koncentrátorom. Na základe analýzy bol vytvorený dátový model pre ukladanie uskutočnených meraní a boli špecifikované technológie použité pre samotnú realizáciu. Ako programovací jazyk bol zvolený PHP vzhľadom k výbornej podpore protokolu SOAP a na ukladanie dát databáza typu PostgreSQL. Realizácia SOAP servera využíva jednoduchý objektový návrh, ktorého cieľom je čo najviac obmedziť nutné zásahy do zdrojového kódu po prípadnej zmene formátu dát posielaných meračmi. SOAP server bol úspešne otestovaný pomocou simulátora a je pripravený na zavedenie do praktického používania.

Návrh informačného systému bol realizovaný pomocou diagramov prípadov použitia a dátového modelu. Na implementáciu bol zvolený PHP framework Nette s využitím architektúry MVC. Hlavný dôraz pri vývoji bol kladený na zabezpečenie a jednoduchú rozširiteľnosť celej aplikácie. V systéme boli vytvorené moduly pre grafické zobrazenie nameraných dát z meračov, administráciu užívateľov a meračov. Používateľské prostredie bolo implementované s použitím frameworku Bootstrap, ktorý zabezpečuje responzívny dizajn informačného systému. Vytvorený systém bol nakoniec odskúšaný s použitím dát spracovaných SOAP serverom a mal by byť schopný obstáť v reálnej prevádzke.

# LITERATÚRA

- [1] INTELIGENTNÉ SIETE [online]. ©2002 – 2015 [cit. 2015-05-18]. Dostupné z URL: <<http://www.seas.sk/smart-grids>>
- [2] International Energy Agency. *Technology Roadmap: Smart Grids*, [online]. France, 2011 [cit. 2015-05-19]. Dostupné z: <[https://www.iea.org/publications/freepublications/publication/smartgrids\\_roadmap.pdf](https://www.iea.org/publications/freepublications/publication/smartgrids_roadmap.pdf)>
- [3] FRANEK, Lešek. *Data koncentrátor pro chytré sítě: diplomová práce*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2012. 114 s. Vedoucí práce Ing. Pavel Kučera, Ph.D.
- [4] Inteligentní sítě – Česká republika nezůstává pozadu [online]. 2010 [cit. 2015-05-18]. Dostupné z URL: <<http://www.ekobydleni.eu/energie/inteligentni-site-ceska-republika-nezustava-pozadu>>
- [5] Evropský kontext [online]. 2010 [cit. 2015-05-18]. Dostupné z URL: <<http://www.futuremotion.cz/smartgrids/cs/index.html>>>
- [6] GOURLEY, David a Brian TOTTY. *HTTP: the definitive guide. 1st ed. Sebastopol*, CA: O'Reilly, 2002, xviii, 635 p. ISBN 15-659-2509-2.
- [7] HTML/Training/What is HTML. [online]. 2013 [cit. 2015-05-18]. Dostupné z URL: <[http://www.w3.org/community/webed/wiki/HTML/Training/What\\_is\\_HTML](http://www.w3.org/community/webed/wiki/HTML/Training/What_is_HTML)>
- [8] HTML & CSS [online]. 2015 [cit. 2015-05-18]. Dostupné z URL: <<http://www.w3.org/standards/webdesign/htmlcss>>
- [9] History of PHP [online]. 2014 [cit. 2015-05-18]. Dostupné z URL: <<http://php.net/manual/en/history.php.php>>
- [10] Your first look at JavaScript [online]. 2012 [cit. 2015-05-18]. Dostupné z URL: <[http://www.w3.org/wiki/Your\\_first\\_look\\_at\\_JavaScript](http://www.w3.org/wiki/Your_first_look_at_JavaScript)>
- [11] Extensible Markup Language (XML) [online]. ©2013 – 2015 [cit. 2015-05-18]. Dostupné z URL: <<http://www.w3.org/XML/>>
- [12] Web Services Description Language (WSDL) [online]. 2011 [cit. 2015-05-18]. Dostupné z URL: <<http://www.w3.org/TR/wsd1>>

- [13] SOAP Version 2 [online].2007 [cit. 2015-05-18]. Dostupné z URL: <<http://www.w3.org/TR/soap12/>>
- [14] About [online]. ©1996 – 2015 [cit. 2015-05-18]. Dostupné z URL: <<http://www.postgresql.org/about/>>
- [15] History [online].2012 [cit. 2015-05-18]. Dostupné z URL: <<http://getbootstrap.com/about/>>
- [16] About jQuery [online]. 2015 [cit. 2015-05-18]. Dostupné z URL: <<https://learn.jquery.com/about-jquery/>>
- [17] Quick Start [online]. ©2008 – 2015 [cit. 2015-05-18]. Dostupné z URL: <<http://dibiphp.com/cs/quick-start>>
- [18] MVC aplikace & presentery [online]. 2015 [cit. 2015-05-18]. Dostupné z URL: <<http://doc.nette.org/cs/2.3/presenters>>
- [19] Zabezpečení před zranitelnostmi [online]. 2015 [cit. 2015-05-18]. Dostupné z URL: <<http://doc.nette.org/cs/2.3/vulnerability-protection>>

## ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

FEPC	The Federation of Electric Power Companies of Japan
PLC	Power line communication
GPRS	General Packet Radio Service
HTTP	hypertext transfer protocol
MIME	Multi-Purpose Internet Mail Extensions
HTML	HyperText Markup Language
W3C	World Wide Web Consortium
CSS	Cascading Style Sheets
PHP	Hypertext Preprocessor
XML	eXtensible Markup Language
SOAP	Simple Object Access Protocol
UML	Unified Modeling Language
WSDL	Web Services Description Language
ORDMBS	Object-Relational Database Management System



# ZOZNAM PRÍLOH

A Obsah priloženého CD

58

## A OBSAH PRILOŽENÉHO CD

- zložka BP s textom tejto práce vo formáte pdf
- zložka WWW s kompletným webovým rozhraním systému
- zložka SOAP s zdrojovými kódmi SOAP servera
- zložka SQL s SQL súborom pre založenie databázy

V súboroch informačného systému je nutné upraviť `www/app-config.local.neon` tak, aby sa systém mohol pripojiť k databázovému serveru. Celý systém bol vyvíjaný a testovaný na webovom servery Apache 2.2.29 s rozšírením PHP 5.6.2 a ako databázový server bol použitý PostgreSQL 9.1.14.